A Framework for Real-Time Physically-Based Hair Rendering

By

Santi Gonzalez Jimenez

Supervisors:
Ricardo Jorge Rodrigues Sepúlveda Marques
José-Angel Blat Gimeno

A thesis submitted for the degree of

Master in Innovation and Research in Informatics
Specialization in Computer Graphics and Virtual Reality

Facultat d'Informàtica de Barcelona
Universitat Politècnica de Catalunya

Barcelona, June 2022

Acknowledgements

I would like to thank the *Grup de Tecnologies Interactives* (GTI)[1] department from *Universitat Pompeu Fabra* (UPF) and each of its members for allowing me to develop this project as a collaboration with them, motivated by the European projects *Present*[2] and *SignON*[3] being carried on by the group. Specifically, I would like to thank my supervisors Dr. Ricardo Marques[4] and Prof. Josep Blat[5] for guiding me through this work and providing very useful feedback regularly.

Also, I want to thank my colleagues from the Huawei IRC 3DE team for the support provided. Specifically, Thomas Poulet[6] for having the patience and expertise to solve many of the technical doubts I've had related to hair rendering, as well as providing helpful feedback during the project development, and Tobias Alexander Franke[7] for introducing me useful insights, initial ideas, and references around the topic.

Finally, I want to thank everyone else involved in my academic and professional career, UPC lecturers, and class colleagues, as well as my family and friends. In this group falls, of course, my cat *Lofi*, who has provided emotional support like no human deserves.

This must have been the most exciting year in my life so far, so again: thank you, everyone, for making it possible.

[1] https://www.upf.edu/web/gti
[2] https://cordis.europa.eu/project/id/856879
[3] https://cordis.europa.eu/project/id/101017255
[4] https://ricjm.github.io/
[5] https://www.upf.edu/web/jblat
[6] https://blog.thomaspoulet.fr/
[7] https://www.tobias-franke.eu/

Abstract

Hair rendering has been a major challenge in computer graphics for several years due to the complex light interactions involved. Complexity mainly stems from two aspects: the number of hair strands, and the resulting complexity of their interaction with light. In general, theoretical approaches towards a realistic hair visualization aim to develop a proper scattering model on a per-strand level, which can be extended in practice to the whole hair volume with ray tracing even though it is usually expensive in computational terms. Aiming at achieving real-time hair rendering, I analyze each component contributing to it from both theoretical and practical points of view in this work. Most approaches, both real- and non-real-time build on top of the Marschner scattering model, such as recent efficient state-of-the-art techniques introduced in Unreal Engine[8] and Frostbite[9], among others. Interactive applications cannot afford the complexity of ray tracing, and they target efficiency by explicitly dealing with each component involved in both single-strand and inter-strand light interactions, applying the necessary simplifications to match the time budget. I have further implemented a framework, separating the different components, which combines aspects of these approaches towards the best possible quality and performance. The implementation achieves real-time good-looking hair, and its flexibility has allowed to perform experiments on performance, scalability, and contribution to quality of the different components.

[8] https://www.unrealengine.com/
[9] https://www.ea.com/frostbite

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Hair rendering has been a challenging field in computer graphics for several years due to its nature. Complexity primarily arises from two aspects: the number of hair strands, and the resulting complexity of their interaction with light. In order to solve these problems, most offline approaches aiming for realistic results are centered on physically-based scattering models per strand. Inter-strand light interactions are usually solved with path tracing by accumulating the per-strand scattered light.

Due to performance constraints, real-time applications cannot benefit from such techniques with such a detailed geometry as the one required for high-quality physically-based hair modeling. Given the complexity of the problem, it is hard for a physically-based hair rendering approach to achieve real-time, and there are not many instances of that in the literature. However, the impact of a realistic hair visualization in any kind of humanoid-based interactive application is very significant, and thus many efforts are being invested into producing real-time solutions for hair rendering. In general, the first real-time approaches introduced were based on empiric scattering models (and thus not photo-realistic). Recently, a new set of prominent techniques stem from simplifications of physically-based models designed for offline rendering, aiming to replicate the behavior of the original models as closely as possible while removing computational complexity from it.

The goal of this work is to explore, analyze, and implement efficient real-time techniques and approximations for physically-based hair rendering and shading.

Most real-time techniques for hair rendering aim to reconstruct an outcome similar to that obtained by offline rendering. Their approach is to decompose the problem into separate rendering components, which are briefly discussed next.

1. **Strand blending**: Hair is a translucent medium, but due to the high density of strands, traditional hardware blending can be slow and not valid. Also, depending on the geometry used to represent the hair, different alternatives are available. In this work, I analyze the options available for blending hair strands represented with hair cards (planes with baked strand information).

2. **Single-strand light scattering**: Several scattering models trying to accurately reproduce the light reflected from a single strand in a specific direction have been introduced. Most of them define a *bidirectional curve scattering distribution function* (BCSDF) for the specific case of a cylindrical translucent fiber. This report discusses the physically and experimentally based Marschner model [MJC+03], which is the basis of most offline, and most recent real-time approaches. The best ones of the latter indeed intend to computationally simplify offline ones, while retaining their key contributions. In this work's implementation, I combine some aspects of such state-of-the-art real-time models, aiming for a good quality-performance ratio.

3. **Environmental lighting**: A proposed real-time model can be discretized for every light source on the scene. However, most applications use environmental lighting to match the appearance of the rendered object within its surrounding environment. For traditional *bidirectional reflectance distribution functions* (BRDF), this is usually done with precomputed diffuse irradiance maps and pre-filtered environment maps. However, when using BCSDFs a more complex integration is required to preserve the mathematical consistency of the rendering equation. This is not feasible in real-time, so this work explores some simplified alternatives.

4. **Self-shadowing**: Because hair is translucent, traditional shadowing techniques cannot be directly applied. Furthermore, the way the strands are blended conditions the shadowing that can be applied. In general, extending traditional techniques to allow a smooth shadow falloff across the volume is efficient and accurate enough, so it is the alternative implemented in this work.

5. **Global light scattering**: To account for the global light scattering occurring inside the hair volume, traditional path tracing can be used by evaluating recursively the chosen reflectance model against the multiple strands hits.

Again, this is computationally complex and usually not feasible even in real-time rendering. Therefore, this work analyzes the actual contribution of this component, and uses a simple and efficient approximation to replace it.

These different components have been integrated into a framework built on top of the emerging open-source 3D engine O3DE[1], with a user interface where components can be enabled or disabled, and parameters adjusted. This allows us to be able to explore and test in detail the different aspects discussed above.

In this report I first introduce some theoretical background in Chapter 2, which is required to understand later chapters. This chapter contains a description of the hair anatomy, the basics of surface reflectance, an explanation of some geometrical representations of hair, and detailed information regarding hair-specific scattering models, emphasizing the Marschner light scattering model [MJC+03], which is the origin of most physically-based models, and indeed most of them build upon it.

The most relevant state-of-the-art for this work is then described in Chapter 3. Afterwards, the actual development of the main work is presented in Chapter 4; first, the proposed framework is introduced, and then the main body of the work is described in detail, discussing the procedures followed to solve the problems previously stated, which in general imply choosing and mixing some of the state-of-the-art techniques, aiming for a good quality-performance ratio. Some results are shown through this chapter as well in order to visually complement the explanations provided, as well as displaying the contribution of each component.

A set of experiments and comparisons is performed in Chapter 5; the implemented model is compared with traditional physically based rendering (PBR) models and evaluated qualitatively. Then, some quantitative experiments are run to measure the cost of each component implemented and the overall scalability.

Finally, this work concludes with a summary of the procedures and results obtained, which have achieved the real-time target, and the potential aspects to be improved in future work. The features of O3DE more relevant to this work are left to an appendix.

[1] https://o3de.org/

Chapter 2

Theoretical Background

This chapter provides the most essential background for later developments. It contains a description of the hair anatomy, the basics of surface reflectance, geometrical representations of hair, and a detailed discussion of hair-specific scattering models, focusing on Marschner's model [MJC+03], which, as indicated in the introduction, is the basis of most other approaches.

2.1 Hair Anatomy

Hair strands are usually approximated as curved elliptical cylinders. Figure 2.1 shows a cross-section visualization of a human hair fiber. As can be observed, a human hair fiber can be split into three major components which are uniform along the fiber length [YJR17]:

- **Cuticle**: Layer surrounding the inner cortex formed by overlapped scales (like a roof), making the fiber appear as a nested set of cones.
- **Cortex**: Pigmented bulk of the fiber that **absorbs** light. It contains most of the colored pigments within the fiber.
- **Medulla**: Pigmented core of the fiber that **scatters** light. The medulla is usually very small for human hair, so its effect can be neglected.

As shown in Section 2.4.2, most hair-specific reflection models build on the human hair decomposition shown in Figure 2.1.

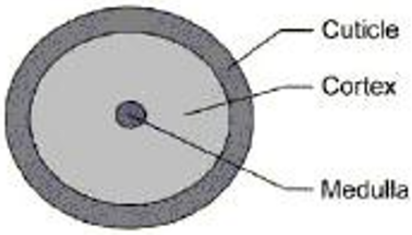


Figure 2.1: Cross-section schematic of human hair fibers [Wei06].

2.2 Surface Reflection

The theory and details introduced in this section are mainly based on the content in Section 5.6 (Surface Reflection) from the book *Physically Based Rendering: From Theory To Implementation* [PJH16].

When a light ray reaches a surface, part of it is absorbed or scattered and the other part is reflected back to the outside. There are two essential components to model the reflected light:

- Spectral distribution: Different materials absorb different wavelengths from the light and therefore will reflect the non-absorbed ones. This property is what defines the color of a material.
- Direction distribution: Due to each material's micro-structure. This property defines the specular lobe width of a material.

To formalize this, some reflection abstractions were introduced in the literature. In this section, I present the BRDF model (which extends to BSDF and BCSDF models used in hair rendering) and the BSSRDF model (most popular for translucent surfaces, used in recent work in the hair rendering literature [YSJR17]).

2.2.1 BRDF

The *bidirectional reflectance distribution function* (BRDF) models reflection at a surface under the assumption that light incident to a surface point will either be absorbed or reflected.
Therefore, the goal is to measure the radiance $L_o(p, \omega_o)$ leaving from a point p in a specific direction ω_o due to the incident radiance $L_i(p, \omega_i)$ at p from a specific direction ω_i. Remark that, for simplicity, we do not make explicit the dependence on the

wavelength. This is visually illustrated in Figure 2.2, where n is the normal vector at the surface point p.

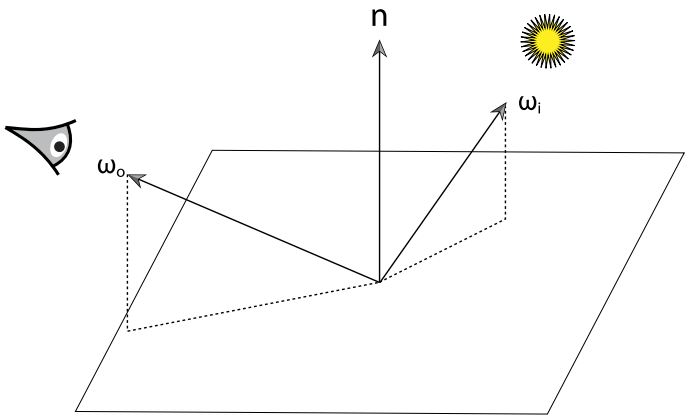


Figure 2.2: Visualization of the BRDF reflectance abstraction [PJH16].

Hence, the BRDF itself is a ratio between both energy quantities. A way to formalize it is by using the irradiance at p. It can be defined as:

$$E(p) = \int_{\Omega} L_i(p, \omega_i) \cos\theta_i \, d\omega_i, \tag{2.1}$$

where Ω is the set of all directions contained in the positive hemisphere around n, and θ_i is the angle between the normal n and the incident direction ω_i.
However, to define the BRDF we are only interested in individual incident directions. Hence, we can extract the differential irradiance $dE(p, \omega_i)$ from the expression above by considering ω_i as a differential cone of directions (usually know as solid angle) with axis ω_i:

$$dE(p, \omega_i) = L_i(p, \omega_i) \cos\theta_i \, d\omega_i \tag{2.2}$$

This differential irradiance contributes differentially to the radiance $L_o(p, \omega_o)$, allowing us to build a ratio defining the BRDF $f_r(p, \omega_o, \omega_i)$:

$$f_r(p, \omega_o, \omega_i) = \frac{dL_o(p, \omega_o)}{dE(p, \omega_i)} = \frac{dL_o(p, \omega_o)}{L_i(p, \omega_i) \cos\theta_i \, d\omega_i} \tag{2.3}$$

From this expression, the outgoing radiance $L_o(p, \omega_o)$ can be defined in terms of the BRDF and the incident radiance:

$$L_o(p, \omega_o) = \int_{\Omega} f_r(p, \omega_o, \omega_i) L_i(p, \omega_i) \cos\theta_i \, d\omega_i \tag{2.4}$$

In the context of physically-based rendering, BRDFs must satisfy the following properties:

- Reciprocity:

$$f_r(p, \omega_o, \omega_i) = f_r(p, \omega_i, \omega_o), \qquad \forall \omega_o, \omega_i \in \Omega \tag{2.5}$$

- Energy conservation: The total radiant exitance cannot be greater than the total energy of incident light.

$$\int_\Omega f_r(p, \omega_o, \omega_i) \cos\theta_i \, d\omega_i \leq 1, \qquad \forall \omega_o \in \Omega \tag{2.6}$$

A trivial variant of the BRDF is the *bidirectional transmittance distribution function* (BTDF) for absorbed light. The form is the same but with ω_i and ω_o belonging to opposite hemispheres. The result of joining the BRDF and the BTDF (which expands to using the whole set of sphere directions as integration domain instead of only the hemispherical ones) is commonly referred to as *bidirectional scattering distribution function* (BSDF). Most hair rendering approaches are based on a specification of the BSDF in the domain of curves, named *Bidirectional Curve Scattering Distribution Function* (BCSDF).

A significant limitation of the BRDF abstraction is that incident light is assumed to be absorbed or reflected at the same point of incidence, while in real-world materials the incident light is scattered at a subsurface level before leaving the surface at a different point (this is partially modelled by a diffuse reflection component of the BRDF) or being absorbed. In most materials, however, the subsurface light transport is negligible, thus the BRDF abstraction is accurate enough. But for translucent materials, this is not the case.

2.2.2 BSSRDF

The *bidirectional scattering surface reflectance distribution function* (BSSRDF) is a generalization of the BRDF that considers scattering from materials that exhibit a significant amount of subsurface light transport.

Similar to the BRDF, the BSSRDF is a distribution function $S(p_o, \omega_o, p_i, \omega_i)$ describing the ratio between outgoing and incident radiance in specific directions. The key difference is that the BSSRDF allows measuring the outgoing radiance at any point p_o that is not necessarily the same point p_i where the incident light arrives, accounting this way for the subsurface scattering contribution. Figure 2.3

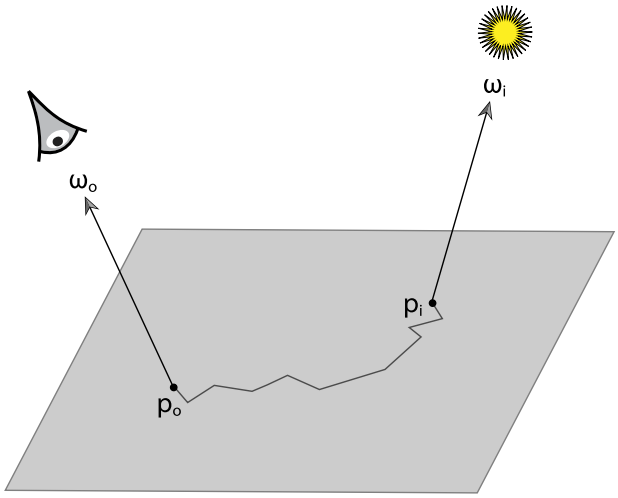


Figure 2.3: Visualization of the BSSRDF reflectance abstraction [PJH16].

describes this abstraction visually. Note that if $p_o = p_i$ the BSSRDF becomes the BRDF introduced above, hence the BSSRDF generalizes the BRDF.

Using the same procedure as in the BRDF, and renaming p the outgoing point as earlier, the BSSRDF can be formalized as:

$$S(p, \omega_o, p_i, \omega_i) = \frac{dL_o(p, \omega_o)}{dE(p_i, \omega_i)} = \frac{dL_o(p, \omega_o)}{L_i(p_i, \omega_i) \cos\theta_i \, d\omega_i} \tag{2.7}$$

Then, the outgoing radiance can be isolated:

$$L_o(p, \omega_o) = \int_A \int_\Omega S(p, \omega_o, p_i, \omega_i) L_i(p_i, \omega_i) \cos\theta_i \, d\omega_i dA \tag{2.8}$$

Note that the inner integral is the same as in Eq. (2.4) describing the BRDF-based radiance. Since now we have another dimension p_i, we need to add another integral to measure the contribution of the incident light at all positions p_i in the surface area A. The rationale behind this is that every point in the surface might contribute with subsurface scattered light to any other point in the surface. In practice, however, the BSSRDF will assign a negligible contribution when p_i and p are far away, since the incident light at p_i is likely to be either absorbed or reflected at other surface points before reaching p.

2.3 Hair Geometry Representation: Cards

Since the pipeline used is based on rasterization, a traditional way to represent hair would be with an explicit geometry. Many explicit representations have been used

Figure 2.4: Hair cards model. Top: shaded hair. Bottom: raw cards geometry. From [Tok18].

in the literature and in interactive applications.
Sometimes a single-shell *mesh* representing the whole hair shape can be used. In other cases, especially for fur, a *shell* by extruding the mesh with strand data has been proposed as a useful representation. The most sophisticated approach uses individual strand data as actual geometry. However, representing hair with hair *cards* is the most common option in current games and interactive applications in general, and thus we focus on it.

In terms of explicit geometry, cards usually refer to planes (curved and elongated) representing a set of strands. The single-strand detail is baked into a texture so that it can be recovered when shading the plane geometry. An instance of hair cards model is shown in Figure 2.4.

There are some significant advantages when using this representation:

- The reduced amount of vertices improves the geometry throughput.
- Reduced amount of overdraw due to the baking strategy (leading to fewer sub-objects per pixel compared to a strand-based representation).

However, despite its advantages, it also exhibits relevant limitations:

- Per-strand detail depends on the texture resolution. For high-quality hair, a high texture resolution is usually needed (2k or 4k).

- Usually multiple textures are required to recreate a proper shading, increasing the memory footprint.
- Hard to blend properly.
- Repeating patterns can be spotted if observed carefully.

Most of these cannot be solved directly but can be alleviated, as done in some parts of this work. Still, cards achieve a very good quality-performance ratio, especially for real-time applications.

2.4 Hair Shading

2.4.1 Near-field and Far-field Scattering

Single fiber scattering can be approximated with two different approaches:

- Far-field scattering models assume that hair fibers are very thin, usually thinner than a pixel. Therefore, these models give more importance to the integral over the hair fiber cross-section rather than the accurate incident position. This implies that incident light across the width of the hair fiber is assumed to be collimated (parallel). Thus, when a strand covers more than one fiber it will look flat.
- Near-field approaches, on the other hand, take into account the accurate incident positions in the cross-section, giving hair a cylindrical appearance from a close-up view.

Figure 2.5 shows a diagram of the scattering in both cases. The full intuition behind it is going to be clearer after going through Section 2.4.2, but a simpler description is presented here. Far-field estimates the angle of incidence of light in the hair cross-section (related to the h parameter in the figure). This estimation is done from the exiting azimuth ϕ we are interested in with respect to the incident light direction. Near-field, on the other hand, considers the actual angle of incidence of the light in the hair cross-section, and computes its contribution to the exiting azimuth ϕ. The figure also shows the different visual appearances resulting from each approach.

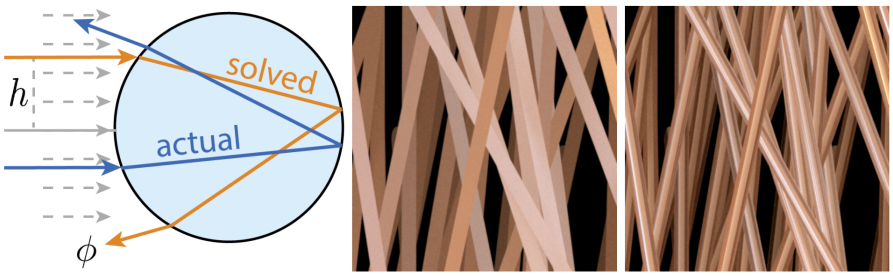


Figure 2.5: Left: (blue) and far (orange) field scattering are represented. Middle: near-field close-up. Right: far-field close-up. [YJR17]

2.4.2 Marschner scattering model

Marschner et al. [MJC+03] introduced one of the first physically-based far-field Bidirectional Curve Scattering Distribution Function (BCSDF, more details in Section 2.4.2.2) models in the hair rendering literature. Most posterior models build upon it. Given its prominence, the model is described in detail in this subsection.

From the hair anatomy described in Section 2.1, Marschner models a hair fiber as an elliptical cylinder with the cuticle being the surrounding layer, and simplifies the cortex and medulla into a single component. Considering an elliptical cross-section introduces a significant complexity both mathematical and computational. Since this work targets real-time, and the elliptical component is not essential for a proper visualization (even though it is not negligible in non-caucasian hairs), a circular cross-section is assumed in this section and over the whole work unless noted otherwise.

2.4.2.1 Notation

Figures 2.6, 2.7 and Table 2.1 show the scattering notation used in their work. The interpretation of these figures is introduced progressively through this section.

The stated notation is also inherited in later work, therefore it is used throughout this work too. As can be observed in Figure 2.6, the scattering and absorption in the interior of the fiber are considered to be uniform, that is, the *cortex* and *medulla* components introduced in Section 2.1 have been merged into a single one.

2.4.2.2 Bidirectional Curve Scattering Distribution Function

Marschner defines a *Bidirectional Curve Scattering Distribution Function* (BCSDF) $S(\omega_i, \omega_o)$, which conceptually maps to a BSDF applied to curves. The difference

Term	Description	Formula
u	Tangent direction to the hair fiber (pointing towards the strand tip)	—
v	Major axis of the elliptical cross section (forming an orthonormal basis with u and w)	—
w	Minor axis of the elliptical cross section (forming an orthonormal basis with u and v)	—
ω_i	Incident light direction	—
ω_r	Direction in which scattered light is being measured	—
$\omega_{x\perp}$	Projection of ω_x onto u (perpendicular to the normal plane v-w)	$u(\omega_x \cdot u)$
$\omega_{x\parallel}$	Projection of ω_x onto the normal plane v-w (parallel to it)	$\omega_x - \omega_{x\perp}$
θ_i	Longitudinal inclination[a] of ω_i	$\arccos(\omega_i \cdot \omega_{i\parallel})$
θ_r	Longitudinal inclination of ω_r	$\arccos(\omega_r \cdot \omega_{r\parallel})$
ϕ_i	Azimuthal inclination[b] of ω_i	$\arccos(\omega_{i\parallel} \cdot v)$
ϕ_r	Azimuthal inclination of ω_r	$\arccos(\omega_{r\parallel} \cdot v)$
θ_d	Difference longitudinal angle	$(\theta_r - \theta_i)/2$
ϕ	Relative azimuth	$\phi_r - \phi_i$
θ_h	Average longitudinal inclination (half angle)	$(\theta_i + \theta_r)/2$
ϕ_h	Average azimuthal inclination (half angle)	$(\phi_i + \phi_r)/2$
α	Cuticle scales tilt	$-10°$ to $-5°$
η	Index of refraction of hair	1.55
η'	Cortex's virtual refractive index in the azimuthal projection	$\frac{\sqrt{\eta^2 - \sin^2\theta_d}}{\cos\theta_d}$
h	Offset from a ray to the center of a unit circle	—
γ_i	Angle of incidence with respect to the cross section circle normal	$\arcsin(h)$
γ_t	Angle of the refracted ray with respect to the cross section circle normal	$\arcsin(h/\eta')$

[a] Inclination with respect to the normal plane.
[b] Inclination around the hair fiber axis, with respect to v.

Table 2.1: Notation description for geometry scattering and hair-specific parameters.

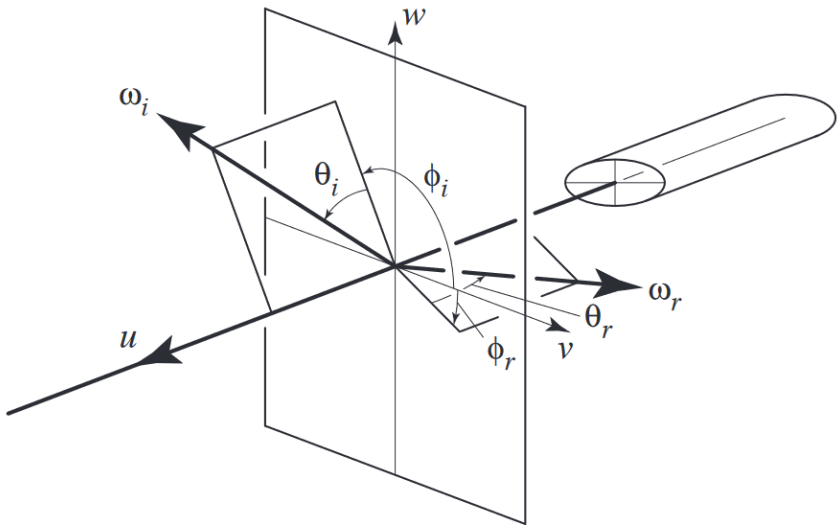


Figure 2.6: Cylinder scattering notation [MJC+03].

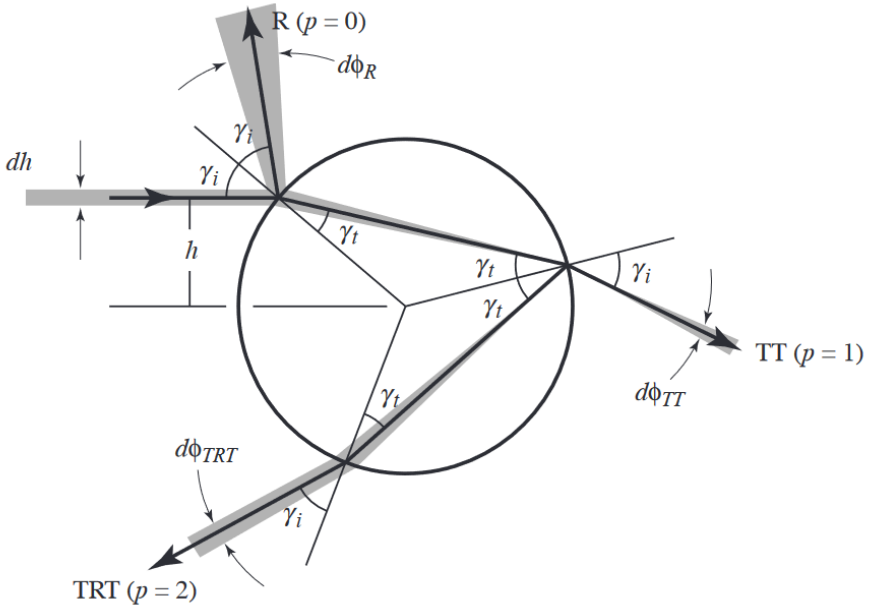


Figure 2.7: Cylinder scattering notation from a circular cross-section [MJC+03].

is that it is measured in terms of curve irradiance $\bar{E}$ (power per unit length) and curve intensity $\bar{L}$ (intensity per unit length), which are analogous to irradiance and radiance, respectively. Using the notation stated in 2.4.2.1, we can write:

$$S(\omega_o, \omega_i) = \frac{d\bar{L}_o(\omega_o)}{d\bar{E}(\omega_i)} = \frac{d\bar{L}_o(\omega_o)}{D\bar{L}_i(\omega_i)\cos\theta_i \, d\omega_i}, \tag{2.9}$$

where D is the diameter of the fiber. Then, the scattering integral is:

$$\bar{L}_o(\omega_o) = D \int S(\omega_o, \omega_i) \bar{L}_i(\omega_i) \cos\theta_i \, d\omega_i \tag{2.10}$$

Qualitatively, the arrangement of D in this expression indicates that a thick fiber gets more light, and therefore appears brighter from a distance than a thin fiber.

2.4.2.3 Components of reflected light for a hair fiber

From their experiments, Marschner et al. [MJC+03] conclude that three scattering paths can be identified as the most relevant ones for translucent hair: R, TT, and TRT, where T and R symbolize *transmission* and *reflection* across the fiber interface, respectively. These three paths can be described as:

- R: Reflection from the fiber surface resulting in a component spread uniformly around a cone. Visually, it is a shift of the primary (imperfect) specular peak toward the root of the strand. According to Marschner, this deviation is hypothesized to be due to the tilt in the scales which exist on the hair fiber.
- TT: Transmission through the hair fiber producing a very bright component that is usually more prominent when looking at it from the opposite side of the hair.
- TRT: Light that reflects on from the inside of the actual hair fiber producing a back-scattering component. Visually, it is a colored secondary peak shifted towards the tip of the strand with respect to the primary specular peak. Sometimes it appears more as a colored fringe on the primary highlight than a separate feature.

Figure 2.8 shows a scheme of these paths and the effect of the cuticle scales according to the Marschner model. In Figure 2.7 the respective light cones are represented.

2.4.2.4 Longitudinal and Azimuthal scattering decomposition

Assuming a symmetric cylinder, the 4D scattering function for a fiber can be factored into a product of two 2D terms: longitudinal scattering M (dependent on the longitudinal inclinations θ) and azimuthal scattering N (dependent on the azimuthal inclinations ϕ).
This is based on two observations:

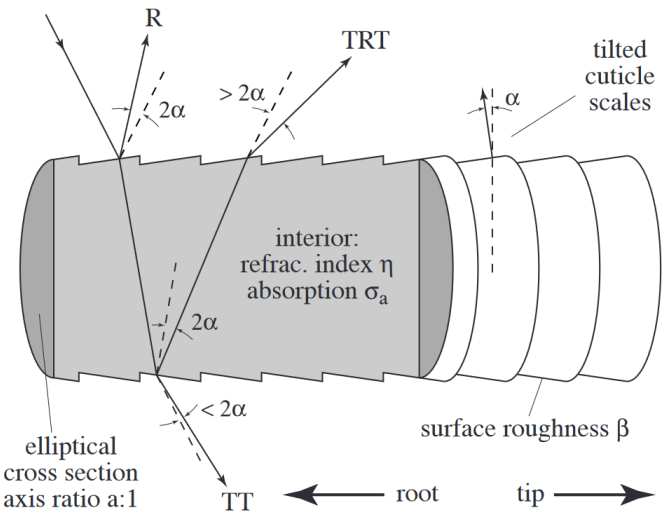


Figure 2.8: Marschner model scheme. The dashed lines indicate the scattering angles for a cylinder without tilted surface scales [MJC+03].

1. In an ideal smooth cylindrical fiber, light arriving at an incoming inclination θ_i is reflected in a restricted specular cone $\theta_r = -\theta_i$ (see Figure 2.9). This also applies to the refracted rays inside the hair, therefore, every path (R, TT, and TRT) involves cone-restricted reflection directions in the longitudinal axis.

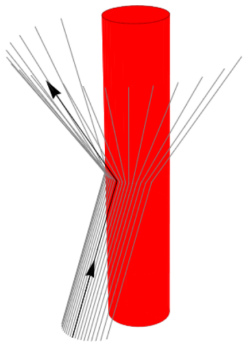


Figure 2.9: Rays reflecting off a smooth cylinder are restricted to a single specular cone [DFH+11].

2. The contribution of the azimuthal scattered distribution can be measured by examining only the projection into a plane perpendicular to the hair fiber.
   For the R path, one can simply project it into the perpendicular plane and apply a traditional reflection using the normal at the incident point, as can be observed in Figure 2.7.
   For the refracted paths, Marschner exploits the Bravais properties of smooth

cylinders, which state that *if the incident and transmitted vectors at a dielectric interface are projected onto a plane containing the surface normal, the projected vectors still obey Snell's law, but with the index of refraction η replaced by $\eta'(\eta, \theta) > \eta$* [MJC+03]. The new index of refraction η' depends on the longitudinal inclination θ. Specifically, $\eta' = \frac{\sqrt{\eta^2 - \sin^2 \theta_d}}{\cos \theta_d}$. The intuition behind this is shown in Figure 2.10. As can be observed, rays with a high incident longitudinal inclination θ_i will appear more refracted from the cross-section projection than rays with no longitudinal inclination $\theta_i = 0$.

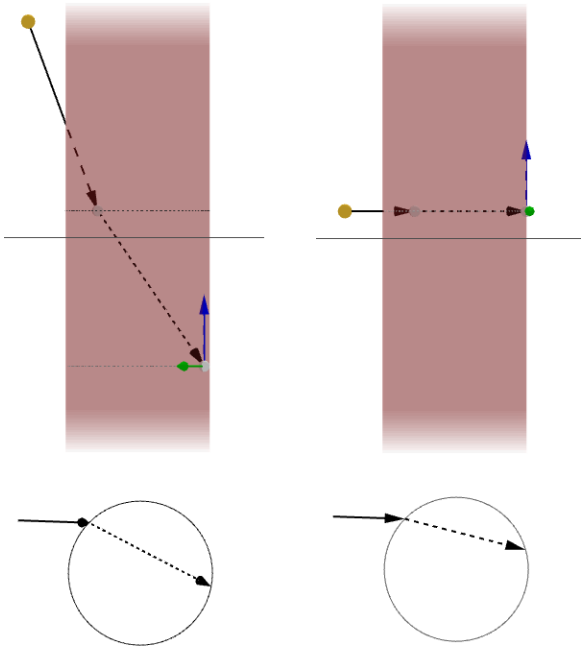


Figure 2.10: Snell's law in a cylinder. (left) high longitudinal inclination ($\theta_i > 60°$) (right) no longitudinal inclination ($\theta_i = 0°$). (top) longitudinal view (bottom) azimuthal view.

On the basis of these two observations, the BCSDF can be decomposed into a longitudinal scattering component $M(\theta_i, \theta_r)$, accounting for the specular cone distribution, and an azimuthal scattering component $N(\eta'(\theta_i); \phi_i, \phi_r)$ performing the general scattering analysis but with a reduced dimensionality:

$$S(\phi_i, \theta_i; \phi_r, \theta_r) = M(\theta_r + \theta_i) N(\eta'(\theta_i); \phi_i, \phi_r) / \cos^2 \theta_d \tag{2.11}$$

The cosine term in this expression accounts for the projected solid angle of the specular cone.

2.4.2.5 Longitudinal Scattering Function

The longitudinal scattering component M would be a simple delta reflection $\delta(\theta_h)$ in the case of an ideally smooth cylinder. However, as seen in the hair morphology (Section 2.1), the exterior layer of a hair strand is constituted by overlapped scales, which are slightly tilted. Due to this, the reflection needs to be offset by this tilt amount. On top of that, the cuticle scales at the surface of a strand are not ideally smooth; some hairs are rougher than others, but it is a significant aspect that must be taken into account. Hence, a normalized Gaussian is used as outgoing distribution, centered at the half-angle θ_h offset by the scales tilt α, and using the roughness amount β as the Gaussian standard deviation.

For every path p (R, TT, TRT), where p is the path index (0, 1, 2) respectively, the tilt α_p and longitudinal roughness β_p to apply are different, since these accumulate with the previous hits. Figure 2.8 and the gray area in Figure 2.7 help to illustrate this. Marschner proposes a practical model using the values shown in Table 2.2.

Tilt values		Roughness values	
α	$-10°$ to $-5°$	β	$5°$ to $10°$
α_R	α	β_R	β
α_{TT}	$-\alpha/2$	β_{TT}	$\beta/2$
α_{TRT}	$-3\alpha/2$	β_{TRT}	2β

Table 2.2: Scales tilt and roughness values used in every path from the Marschner practical model [MJC+03].

2.4.2.6 Azimuthal Scattering

It is not straightforward to measure the scattering distribution for every path p (R, TT, TRT).
Looking at Figure 2.7, we can check that an incident ray deviates by $-2\gamma_i$ for an external reflection, $\gamma_t - \gamma_i$ when entering and leaving the circle, and $\pi + 2\gamma$ for every internal reflection. Hence, the net change in azimuthal direction ϕ for each path is:

- $\phi_R = -2\gamma_i$: a single external reflection occurs.
- $\phi_{TT} = 2\gamma_t - 2\gamma - i + \pi$: the ray enters and leaves the circle. π accounts for the reversal in direction from ω_i, since it points outwards.

- $\phi_{TRT} = 4\gamma_t - 2\gamma - i + 2\pi$: the ray enters and leaves the circle, plus it performs an internal reflection. The same direction reversal as in ϕ_{TT} is applied in this case for the same reason.

This can be put together into a single expression in terms of the path p and the offset $h \in \{-1, 1\}$ from the circle center:

$$\phi(p, h) = 2p\gamma_t - 2\gamma_i + p\pi \tag{2.12}$$

Where $\sin\gamma_i = h$ and $\eta' \sin\gamma_t = h$. The Marschner model measures far-field scattering, meaning that it aims to estimate the offset h given ω_i and ω_o, in order to measure its contribution without evaluating the intersection with any explicit cylinder. Hence, knowing the actual difference azimuthal angle ϕ (see Table 2.1), Marschner finds h by solving the roots of the function $\phi(p, h) - \phi = 0$. In the R ($p = 0$) and TT ($p = 1$) paths there is only one root, while in the TRT ($p = 2$) there might be one or three, meaning that in some scenarios, multiple incident directions γ_i can produce the same output diretion ϕ. The solutions for this system are denoted $h(p, r, \phi)$ for the root r given path p and ϕ.

With this expression, the intensity of the scattered light can be measured by using the principle of energy conservation: *when the curve irradiance $\bar{E}$ illuminates a fiber, uniform irradiance $E(h) = \bar{E}/2$ falls on the cross-section across its width.*
With this, and assuming a negligible attenuation inside the fiber, the exitant intensity in an angular interval $d\phi$ can be measured from the curve irradiance from a small interval dh in the incident direction:

$$\begin{aligned} \bar{L}(\phi(h))d\phi = E(h)dh = \frac{\bar{E}}{2}dh \\ \bar{L}(\phi(h)) = \left|2\frac{d\phi}{dh}\right|^{-1} \bar{E} \end{aligned} \tag{2.13}$$

2.4.2.7 Attenuation by absorption and reflection

In the previous section, the attenuation was assumed to be negligible for the sake of the explanation, but in real life, the attenuation and absorption in the cortex are essential contributors to the color of hair.
The attenuation is measured in terms of the path length. Applying trigonometry, the azimuthal length of each internal path is $2 + 2\cos(2\gamma_t)$ times the radius of the fiber.

The volumetric absorption in the cortex/medulla can be then measured with an absorption parameter σ_a (per unit lenght, which in this case is the radius of the fiber). Hence, each internal path is weighted by its corrsponding absorption $T(\sigma_a, h) = exp(-2\sigma_a(1 + \cos(2\gamma_t))$.
The R component has no absortion since there is no internal path. The TT component has one internal path, so $T_{TT} = T(\sigma_a, h)$. The TRT component has two internal paths, so the corresponding path absorption is applied twice $T_{TRT} = T(\sigma_a, h)^2$.

The other aspect involved in the attenuation computation is the Fresnel reflection. It is important to properly measure the proportion of light that is reflected/refracted for each internal path. With this taken into account, the attenuation expression is defined as:

$$A(0, h) = F(\eta, \gamma_i)A(p, h) = (1 - f)^2 f^{p-1} T(\mu_a, h)^p \tag{2.14}$$

where f is the Fresnel reflection accounting for the longitudinal contribution (based d'Eon et al. [DFH+11] formulation for the sake of simplicity):

$$f = F(\eta, \cos\theta_d \cos\gamma_i) \tag{2.15}$$

Note that for the R path (with index $p = 0$) there's only need to evaluate the reflected proportion of light as usual, since it is indeed a single reflection contribution. In order to understand properly the chained Fresnel reflections from Eq. (2.14) for the other paths, let us trace the TT path attenuation due to a ray perpendicular to the fiber main axis, so that $\theta_d = 0$, (matching the view from Figure 2.7), and a cross-section inclination γ_i:

1. The hair fiber is intersected in an inclination of γ_i with respect to its azimuthal circle normal. The first hit is transmitted. Since the Fresnel models the proportion of light reflected, the transmitted component is $1 - F(\eta, \cos\gamma_i)$.

2. The next hit is transmitted too. Since the input and output medium have swapped, we would need to invert the index of refraction and use the new incident angle γ_t $(1 - F(1/\eta, \cos\gamma_t))$; but because of the Fresnel inter-medium symmetry, we can state that $1 - F(1/\eta, \cos\gamma_t) = 1 - F(\eta, \cos\gamma_i)$, if γ_i is the refracted direction from the inside to the outside of the fiber, which we know is the case.
   Then, combining both hits together we get $(1-f)^2$, which matches Eq. (2.14).

(3). If we wanted to consider the TRT path, the same reasoning would apply, but with an additional reflected hit $F(\eta, \gamma_i)$, for a total contribution $(1-f)^2 f$, matching again Eq. (2.14).

Back to the 3D cylinder, as soon as the longitudinal angle θ_d increases, the reflected components increase, and consequently the transmitted ones decrease.

Putting this together with the principle of energy conservation stated in Eq. (2.13), the total azimuthal scattering N can be defined as:

$$N(\phi) = \sum_p N_p(p, \phi) \tag{2.16}$$

$$N_p(p, \phi) = \sum_r A(p, h(p, r, \phi)) \left| 2\frac{d\phi}{dh}(p, h(p, r, \phi)) \right|^{-1} \tag{2.17}$$

This allows concluding that the contribution of every root r in a path p can be measured as the product of an attenuation function and a distribution function (left and right-hand terms in the summation, respectively).

Chapter 3

State of the Art

This chapter goes through the major contributions of recent scattering models introduced in the literature and other lighting components involved in hair rendering.

3.1 Offline Scattering Models

D'Eon et al. [DFH+11] introduce an extension of Marschner's model [MJC+03] that ensures energy conservation in the longitudinal specularities, and also accounts for cross-section roughness β_N additionally to the longitudinal one (β_M, introduced as β in Section 2.4.2.5). On top of that, d'Eon et al. also use another path, TRRT. It is a far-field model, and their evaluation uses Gaussian quadrature and Taylor expansions. It achieves better results than Marschner but introduces more complexity.

Yan et al. [YTJR15] introduce a double cylinder fur reflectance model, based on hair anatomy, modeling each strand with cuticle-cortex-medulla, each one with a different index of refraction, so that multiple light interactions need to be simulated. It is a near-field model. The effect of the medulla is almost negligible in human hair, so this kind of models usually target animal fur.

Chiang et al. [CBTB16] introduce a practical near-field formulation avoiding the expensive integral across the fiber by distributing azimuthal roughness in a closed form based on a logistic distribution.

Later, Yan et al. [YJR17] propose an efficient extension of their previous work [YTJR15] that simplifies the model to only 5 lobes.

3.2 Shadowing

Yuksel et al. [YK08] introduce a deep opacity maps method, which is an extension of opacity shadow maps [KN01] that achieves semi-transparent shadows adapting to the shape of the hair.

3.3 Global Scattering

Zinke et al. [ZYWK08] introduce a dual scattering method to measure the global scattering inside the hair volume by relying on BCSDFs, specifically using a simplification that decomposes it into forward and backward lobes, which allows them to accumulate light easily in the most important directions. This method can be optimized with a spatial structure, or with deep opacity maps [YK08], but it still involves an additional cost.

Yan et al [YSJR17] introduce the first global illumination model based on dipole diffusion for subsurface scattering (BSSRDF) that approximates light bouncing between individual fur fibers. With this model, light-hair interactions can be modelled with subsurface scattering and a specific set of scattering parameters that they extract from the fiber properties using a neural network.

3.4 Real-time pipelines

The models described above obtain physically accurate results in general. Some achieve interactive framerates but do not implement the other components contributing to a proper hair visualization (described in the Introduction (1)). Given the significance of a realistic hair visualization in any kind of human-based interactive application, a lot of efforts have been invested into creating real-time solutions for hair rendering. Some are based on heuristics, but recently the most prominent ones are simplifications of the models introduced above, trying to replicate as much as possible the behavior of the reference model. This work is strongly based on this kind of references due to the real-time target. Some of these are described in the following subsections.

3.4.1 Epic Games

Karis [Bri16] presents the way hair rendering is handled in the 3D engine *Unreal Engine*[1]. Several simplifications of the Marschner [MJC+03] and d'Eon [DFH+11] scattering models are performed. No shadowing technique from the hair literature is used. Instead, they use an exponential falloff, which achieves good enough results. As for the environmental lighting, the precomputed environment irradiance map is sampled towards a bent normal direction, and the sample obtained is input into the scattering model with some modifications. For global scattering, a fake normal and an exponential falloff towards the shadow map is used again, but this time applying absorption over the light path.

3.4.2 Frostbite

Tafuri [Taf19] explains how their 3D engine *Frostbite*[2] manages hair rendering. An important difference with respect to Karis [Bri16] is that strand-based hair models are used. He presents how Frostbite manages sampling the hair strands efficiently and without artifacts by using a visibility buffer. Also, the scattering model from Karis is extended to depend on azimuthal roughness as well, and some specific components are slightly modified. As for shadowing and global scattering, they use deep opacity maps [YK08].

[1] https://www.unrealengine.com/
[2] https://www.ea.com/frostbite

Chapter 4

Implementation and Results

4.1 Framework set-up and overview

As indicated in the introduction, this work implements a model for shading hair defined by a set of components: hair blending, scattering model, environmental lighting, shadowing, and global scattering, which are discussed in detail in the different sections of this chapter. The implementation introduced in this thesis is a framework integrating these different components, which can be enabled, disabled, or parameterized through a graphical user interface. In this section, I provide an overview of the framework.

4.1.1 Rasterization Pipeline

The framework developed uses a rasterization pipeline, which is a common option in most interactive 3D applications. This is relevant because optimizations of the shading models can differ according to the pipeline used.

The most intuitive pipeline for realistic rendering, in general, would be a traditional ray-tracing pipeline since it allows easily replicating the physical behavior of light; the light reaching the camera can be computed by spawning rays from it and evaluating the BCSDF for every hair strand intersected, but in order to account for the incident light from every direction, more rays need to be spawned from the intersection points, repeating this process recursively. If an infinite level of recursion was allowed, the behavior of light would be accurately represented. Therefore, a ray-tracing pipeline implicitly accounts for hair self-shadowing, translucency, and

global scattering through the volume.

A rasterization pipeline, on the other hand, involves projecting 3D models into the camera view space, and then shading each pixel according to the attributes and contextual information of these models. Therefore, each point being shaded does not have accurate information regarding the light reaching it. This forces us to deal explicitly with the global phenomena mentioned above (self-shadowing, strand blending, and light scattering through the volume). In a rasterization pipeline the light rays cannot be easily simulated, so this usually involves not matching the accuracy of a ray-traced result. Despite this, because of the way GPUs are built and the computational complexity of each approach, a rasterization pipeline is much more efficient and feasible for real-time applications.

4.1.2 Material, Shaders and Geometry

I have been using a new custom material for hair using *O3DE*[1] as 3D engine. *O3DE* is an emerging open-source project which offers high modularity.

O3DE material Briefly speaking, an *O3DE* material is defined as a JSON file that contains a set of properties (such as albedo, roughness, etc), their expected input data type, and how they map to the actual shaders attached to the material. For more information about *O3DE* and its renderer *Atom*, see Appendix A.

Custom shaders In this project, I have implemented three custom shaders: two of these are attached to the depth and shadow pass, which, as the following sections will show, are both necessary to perform proper blending and shadowing. The other one defines the main behavior for the *forward pass* (O3DE uses a *forward+* pipeline), gathering results from the information produced in the previous passes and evaluating the proposed scattering model from this information and the input parameters defined.

[1] https://o3de.org/

Framework interface The properties defined in the material file are shown in the O3DE's material editor as an interface to the actual values used by the shaders. This provides higher flexibility and a more intuitive tuning of the model parameters. Most properties map directly to the forward pass shader, but some others require some kind of processing before being usable, which is performed in *Lua*/*C++* functors (referenced in the material file as well). Besides the parameters strictly required by the shading model, I have added supplementary properties allowing for each component to be enabled/disabled, or in some cases, to be weighted by a user-defined factor. Figure 4.1 shows the framework interface with the available properties in the *Material Property* tab.

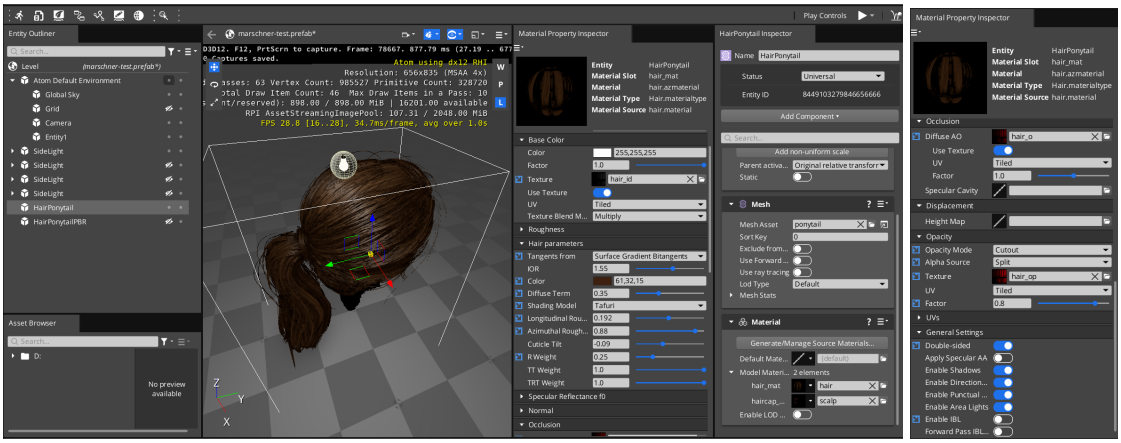


Figure 4.1: Left: framework interface showing the relevant properties exposed for tuning the hair shading model. Right: more properties that are not shown in the left image.

Hair geometry The most popular representations of hair geometry in the context of interactive applications are *cards-based* models and *strand-based* models. In general, cards models are less expensive to shade (see Sections 2.3), so these are the ones used in this work.

4.1.3 Base Scenario

In the following sections, each component involved in the shading model is described in detail. Moreover, the contribution of each one is illustrated with images coming directly from this work's results. The base scenario is the one shown in

4.2, using as default roughness values $\beta_M = 0.2$ (longitudinal roughness, see Section 2.4.2.5) and $\beta_N = 0.8$ (azimuthal roughness introduced first by d'Eon et al [DFH+11], see Section 3.1). The scene setup has been kept as simple as possible in order to minimize noise from other sources, therefore it consists of a single point light that illuminates the hair object.

Figure 4.2: Base sample using the proposed scattering model with shadows enabled.

4.2 Blending

This work uses hair cards models to represent the hair geometry. With proper models of this kind, a plausible visualization can be achieved, but we need to perform blending carefully, where blending refers to the process of combining the contribution from multiple objects (strands, in this case) into the same projected area (pixel) due to their translucent nature. One of the biggest challenges when using hair cards is that blending the cards properly can be cumbersome due to the very thin hair strand width and its translucent nature, as detailed in the next subsections.

4.2.1 Alpha blending

Assuming we are provided with an opacity map, we still cannot perform alpha blending properly because in the case of self-occluding objects like hair cards,

hardware blending does not guarantee any depth order, so we would end up with a corrupted blending as we see next:

- With **enabled depth writes**: when a card is drawn, its depth is written into the depth buffer. For cards drawn afterwards, all fragments falling behind the closest depth registered in these pixels will be discarded, even if the fragment that defined the closest depth (occluder) was transparent.
- With **disabled depth writes**: the cards visible depend on the order they are drawn, so cards that should be occluded can appear completely visible.

4.2.2 Order Independent Transparency

A potential solution for card blending is to use Order-Independent Transparency (OIT) techniques, which consist of sorting the depth of the geometry per pixel in order to blend from furthest to nearest, solving this way the shortcomings of both alpha blending options introduced in the previous section.

However, OIT usually involves storing all fragments in an intermediate step before the sorting and blending stages. As can be expected, these aspects imply a very complex implementation process and a significant overhead in computational cost. Hence, given the real-time target of this work, OIT has been discarded as a blending solution for it.

4.2.3 Alpha testing

The most efficient solution in these scenarios is to perform alpha testing. Alpha test writes into the depth buffer, but only for fragments whose opacity is above a certain threshold.

Naturally, it has some side effects:

- Only the closest fragment (the fragment with lowest depth value) will be shown; there is no blending with fragments behind.
- Sharp aliased edges are usually introduced due to the differences in opacity values of neighbor texels, unless a very high resolution texture is used. Even with post-processing using anti-aliasing techniques, jagged artifacts will still appear when the object is close enough.

- It is hard to determine the proper threshold because the thickness of the strands can be affected. A higher threshold will allow more texels to be displayed and therefore increase the apparent strand width.

Despite this, for the sake of reducing render times, I use this as blending solution. As an important side note, hair cards should be seen from both front and back directions, so face culling needs to be disabled. Figure 4.3 shows some rendering results with different blending modes and threshold values. We can observe in Fig. 4.3(a) that treating the cards as opaque objects does not achieve the desired effect. As stated in this Section 4.2, alpha blending does not ensure a proper blending, as we can see in Figure 4.3(b), specifically in the base hair cap. Still, some arbitrary parts of the ponytail seem to be blended properly. Then, the outcome of using alpha testing with three different threshold values is shown in Figs. 4.3(c) to (e). With a low threshold (Fig. 4.3(c)), some discontinuities become too obvious, apart from the jagged artifacts. With a high threshold (4.3(d)), the strands appear too thick. Figure 4.3(e) shows the threshold value used for the final result, which achieves a good trade-off between both options.

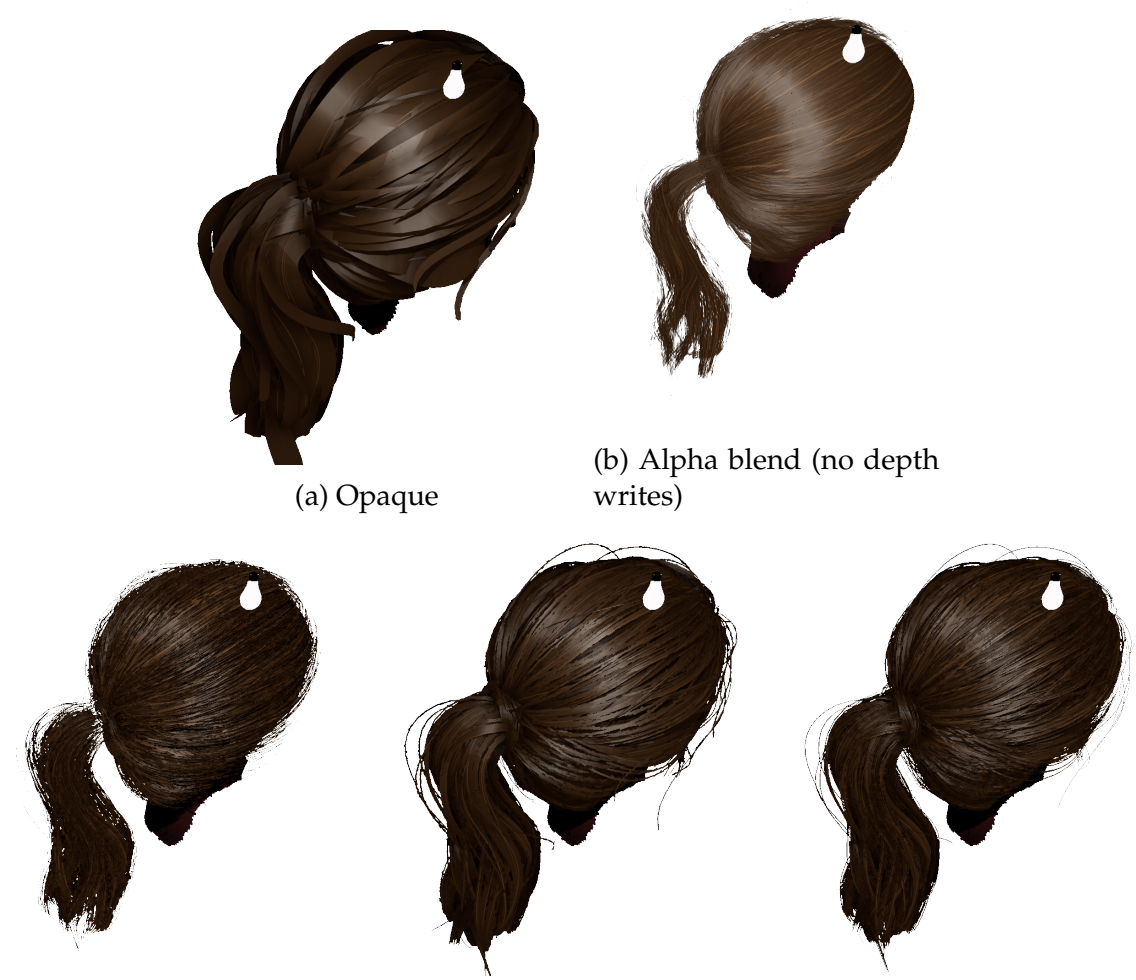
There exist a couple extensions to alpha test that would improve the results at the cost of a slight increase in computational complexity, but have not been implemented due to time constraints:

- **Alpha to coverage** [Gol17]: the goal of this technique is to obtain anti-aliased alpha test. It works by performing multi-sampling and evaluating alpha test for all the coverage samples of the pixel. If a fragment is opaque, then its depth will be written and the color information belonging to that fragment updated. This way, an average of the strands contributing to that pixel will be achieved instead of a unique comparison leading to aliased edges.

- **Hashed alpha testing** [WM17]: this technique makes use of well-distributed noise with spatial and temporal stability to define the alpha test threshold. This kind of noise is obtained with hash functions ensuring a set of spatio-temporal properties (see reference for further information). Hashed alpha testing is usually combined with temporal anti-aliasing.

4.3 Scattering model

The light scattering model is the rendering component that most contributes to a realistic hair visualization. Therefore, a significant part of the efforts involved in



(a) Opaque

(b) Alpha blend (no depth writes)

(c) Alpha test (thr = 0.25)

(d) Alpha test (thr = 1)

(e) Alpha test (thr = 0.8)

Figure 4.3: Outcome of applying different blending modes and parameters.

this project has been devoted to implementing a proper BSDF model, by integrating different state-of-the-art approaches, which we reference in what follows.
The main idea is to model single strand light scattering with three lobes, as introduced by the Marschner BSDF [MJC+03] (see Section 2.4.2), but reducing the computation complexity of each one through mathematical approximations and/or computational optimizations integrating different contributions. First, we recall the basics and notation of Marschner's model and specify the issues dealt with, then discuss more precisely the approximation taken for the different paths (R, TT, TRT) and conclude with a specific subsection on roughness, an important tunable parameter in PBR.

4.3.1 Recall of Marschner's model and Notation

For the reader convenience, Table 4.1, Figure 4.4, Figure 4.5, and Figure 4.6, are inserted again in this section.

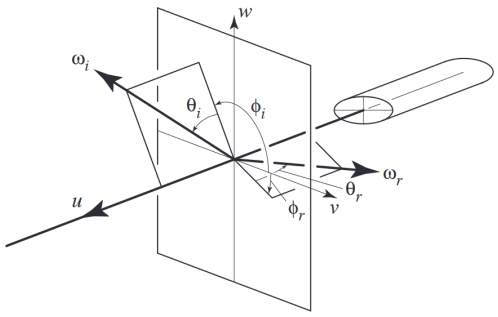


Figure 4.4: Cylinder scattering notation [MJC+03].

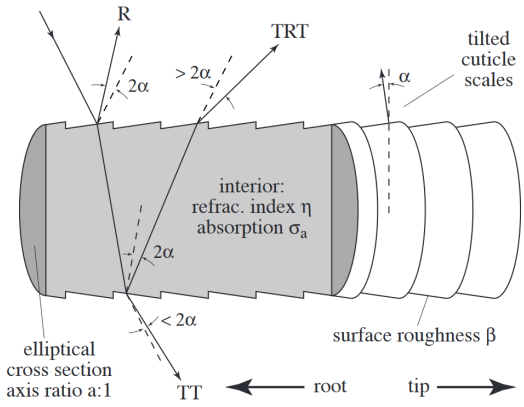


Figure 4.5: Marschner model scheme. The dashed lines indicate the scattering angles for a cylinder without tilted surface scales [MJC+03].

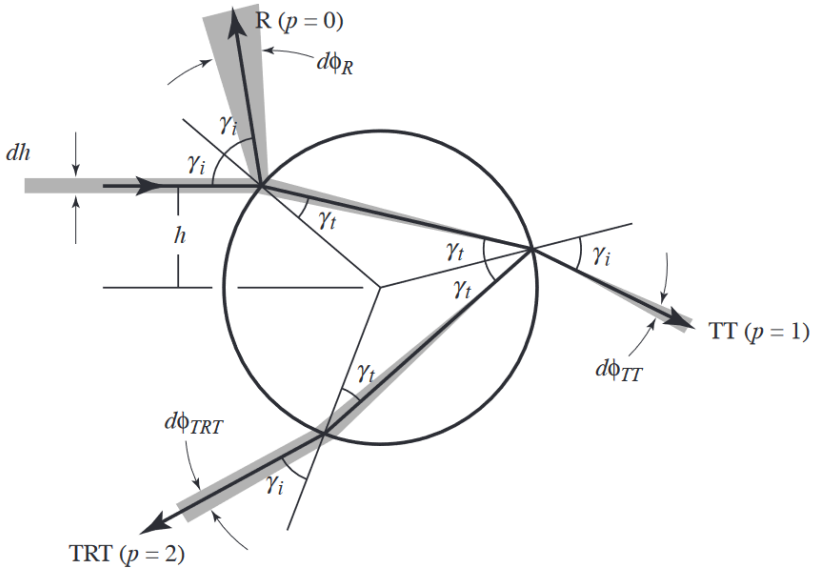


Figure 4.6: Cylinder scattering notation from a circular cross-section [MJC+03].

Marschner et al. [MJC+03] model a hair strand as a translucent cylinder. It measures its BCSDF S as the product of two separate components: longitudinal scattering M and azimuthal scattering N, yielding:

$$S(\phi_i, \theta_i; \phi_r, \theta_r) = M(\theta_i, \theta_r) N(\eta'(\eta, \theta_d); \phi_i, \phi_r) / \cos^2 \theta_d \tag{4.1}$$

Term	Description	Formula
u	Tangent direction to the hair fiber (pointing towards the strand tip)	—
v	Major axis of the elliptical cross section (forming an orthonormal basis with u and w)	—
w	Minor axis of the elliptical cross section (forming an orthonormal basis with u and v)	—
ω_i	Incident light direction	—
ω_r	Direction in which scattered light is being measured	—
$\omega_{x\perp}$	Projection of ω_x onto u (perpendicular to the normal plane v-w)	$u(\omega_x \cdot u)$
$\omega_{x\parallel}$	Projection of ω_x onto the normal plane v-w (parallel to it)	$\omega_x - \omega_{x\perp}$
θ_i	Longitudinal inclination[a] of ω_i	$\arccos(\omega_i \cdot \omega_{i\parallel})$
θ_r	Longitudinal inclination of ω_r	$\arccos(\omega_r \cdot \omega_{r\parallel})$
ϕ_i	Azimuthal inclination[b] of ω_i	$\arccos(\omega_{i\parallel} \cdot v)$
ϕ_r	Azimuthal inclination of ω_r	$\arccos(\omega_{r\parallel} \cdot v)$
θ_d	Difference longitudinal angle	$(\theta_r - \theta_i)/2$
ϕ	Relative azimuth	$\phi_r - \phi_i$
θ_h	Average longitudinal inclination (half angle)	$(\theta_i + \theta_r)/2$
ϕ_h	Average azimuthal inclination (half angle)	$(\phi_i + \phi_r)/2$
α	Cuticle scales tilt	$-10°$ to $-5°$
η	Index of refraction of hair	1.55
η'	Cortex's virtual refractive index in the azimuthal projection	$\frac{\sqrt{\eta^2 - \sin^2\theta_d}}{\cos\theta_d}$
h	Offset from a ray to the center of a unit circle	—
γ_i	Angle of incidence with respect to the cross section circle normal	$\arcsin(h)$
γ_t	Angle of the refracted ray with respect to the cross section circle normal	$\arcsin(h/\eta')$

[a] Inclination with respect to the normal plane.
[b] Inclination around the hair fiber axis, with respect to v.

Table 4.1: Notation description for geometry scattering and hair-specific parameters.

Furthermore, they consider that three scattered paths are enough to reproduce the proper reflectance. These paths are defined as R, TT, and TRT, where T stands for transmission, and R stands for reflection. Each path shape can be better seen in Figure 4.5. According to Marschner et al. [MJC+03], the BCSDF S stated in Eq. (4.1) can then be rearanged in terms of the scattering contribution from each path:

$$\begin{aligned} S(\phi_i, \theta_i; \phi_r, \theta_r) =& M_R(\theta_h) N_R(\eta'(\eta, \theta_d); \phi_i, \phi_r) / \cos^2 \theta_d + \\ & M_{TT}(\theta_h) N_{TT}(\eta'(\eta, \theta_d); \phi_i, \phi_r) / \cos^2 \theta_d + \\ & M_{TRT}(\theta_h) N_{TRT}(\eta'(\eta^*(\phi_h), \theta_d); \phi_i, \phi_r) / \cos^2 \theta_d \end{aligned} \tag{4.2}$$

This work's implementation allows setting a specific longitudinal and azimuthal roughness for hair strands, β_M and β_N, respectively.

Marschner et al. [MJC+03] define the longitudinal lobes M_p as Gaussian functions, aiming to widen the distribution of outgoing directions (hair is not perfectly smooth) according to the longitudinal roughness β_p, and shifting the reflections involved due to the cuticle scales tilt α_p (see Figure 4.5). Evaluating these Gaussian functions is not computationally expensive, therefore there is no need to further simplify the longitudinal lobes definitions:

$$\begin{aligned} M_R(\theta_h) &= g(\beta_R; \theta_h - \alpha_R) \\ M_{TT}(\theta_h) &= g(\beta_{TT}; \theta_h - \alpha_{TT}) \\ M_{TRT}(\theta_h) &= g(\beta_{TRT}; \theta_h - \alpha_{TRT}) \end{aligned} \tag{4.3}$$

The longitudinal roughness β_p and cuticle scales tilt α_p values used in each path are stated in Table 2.2. $g(\sigma, x)$ evaluates x in a normalized zero-mean Gaussian function with standard deviation σ.

Accurately evaluating the azimuthal lobes is complex. Marschner et al. [MJC+03] assume an ideally smooth azimuthal section, which allows to find analytic solutions.

Ultimately, we are interested in finding the offset h of the incident ray with respect to the strand center ($h \in \{-1, 1\}$) because it will allow us to evaluate properly the angle of incidence $\gamma_i = \arcsin(h)$ and the refraction angle $\gamma_t = \arcsin h/\eta'$ in the cross-section of the actual cylinder, which is required in order to reproduce the contribution of the incident light to the scattering in the output direction ϕ_r. Marschner et al. [MJC+03] defines the azimuthal difference angle ϕ for a given path p and incident offset h as:

$$\phi(p, h) = 2p\gamma_t - 2\gamma_i + p\pi \tag{4.4}$$

To understand the intuition behind this expression, see Section 2.4.2.6. In the case of a far-field approximation of a cylinder, we cannot know at which part of the azimuthal section ω_i and ω_r enter or leave, respectively. However, as seen in Section 2.4.2.6, there is only one possible value h for every pair (ω_i, ω_o) for the R and TT paths, and one or three for the TRT path. This is achieved by finding the roots for h of $\phi(p,h) - \phi = 0$.
In practice, ϕ can be obtained easily from ω_i and ω_r (see Table 2.1).

From Marschner et al. [MJC+03] then, the complete normal-plane scattering function N is:

$$N(\phi) = \sum_p N_p(p,\phi) \tag{4.5}$$

$$N_p(p,\phi) = \sum_r A(p,h(p,r,\phi)) \left| 2\frac{d\phi}{dh}(p,h(p,r,\phi)) \right|^{-1} \tag{4.6}$$

As can be observed, the contribution of every root r in a path p can be measured as the product of an attenuation function and a distribution function following the principle of energy conservation stated in Eq. (2.13)(left and right-hand terms in the summation, respectively). These terms are explained in detail in Section 2.4.2.6 and 2.4.2.7.

The attenuation is decomposed into a Fresnel term accounting for the reflected or refracted portions of light, and an absorption term defining the light absorbed in terms of the distance travelled by the light for each path. Since each path has different interactions with the strand, the attenuation A_p and absorption T_p are defined differently for each one. The R path is just a reflection, so its attenuation is just the Fresnel term accounting for the amount of light reflected, with no absorption. The other paths need to consider the multiple surface hits in both the Fresnel and absorption components. Hence, the absorption contribution for each path can be put together in the following expressions:

$$A(0,h) = F(\eta,\gamma_i)A(p,h) = (1-f)^2 f^{p-1} T(\mu_a,h)^p \tag{4.7}$$

where f is the Fresnel reflection accounting for the longitudinal contribution (using d'Eon et al. [DFH+11] formulation for the sake of simplicity):

$$f = F(\eta, \cos\theta_d \cos\gamma_i) \tag{4.8}$$

Again, a more in-depth interpretation of these terms is presented in Section 2.4.2.7.

The longitudinal scattering contribution M_p of every path p is straightforward to compute from Eq. (4.3) and Table 2.2, so the following sections focus on the azimuthal scattering contribution N_p of every path p.

4.3.2 R Path

In order to solve the azimuthal scattering due to the R path N_R, the strategy is to develop each term in Eq. (4.6). Both the attenuation A and the distribution (right-hand term) depend on h, so let us solve it by evaluating $\phi(0,h) - \phi = 0$, where $\phi(p,h)$ is defined in Eq. (4.4) (R path has index $p = 0$). For the R path then, $\phi = -2\gamma_i$. Knowing that $\gamma_i = \arcsin(h)$:

$$\begin{aligned} -2\gamma_i - \phi &= 0 \\ \gamma_i &= -\frac{\phi}{2} \\ h &= -\sin\frac{\phi}{2} \end{aligned} \tag{4.9}$$

In this case, the solution has a single root and only depends on the angle of incidence γ_i, so it is easy to solve analytically.

Most of the following operations requiring γ_i are cosines. Since $\cos -\gamma_i = \cos\gamma_i$, the negative sign will be automatically removed in these cases.

Then, we can develop the distribution term $\left|2\frac{d\phi}{dh}(p,h(p,r,\phi))\right|^{-1}$ in the scattering function N_R from Eq. (4.6) by taking the derivative of the net change in the azimuthal direction $\phi(p,h)$ with respect to h, again, considering $\phi(0,h) = -2\gamma_i$ for the R path, and $\gamma_i = \arcsin(h)$:

$$\begin{aligned} \left|2\frac{d\phi}{dh}(p,h(p,r,\phi))\right|^{-1} &= \left|2\frac{d(-2\gamma_i)}{dh}\right|^{-1} = \left|2\frac{d\left(-2\arcsin h\right)}{dh}\right|^{-1} = \\ \left|2\frac{-2}{\sqrt{1-h^2}}\right|^{-1} &= \frac{1}{4}\left|\sqrt{1-\sin^2\gamma_i}\right| = \frac{1}{4}\left|\cos\gamma_i\right| = \frac{1}{4}\left|\cos\frac{\phi}{2}\right| \end{aligned} \tag{4.10}$$

The attenuation term defined by Marschner uses a 3D generalization of the Fresnel with the Bravais index. Later work shows that this generalization is unnecessary [DFH+11], so we can avoid computing the supplementary parameters

related to it. Hence, the attenuation A for the R path is defined as:

$$A(0,h) = F\left(\eta, \frac{\omega_i \cdot \omega_r}{2}\right) \tag{4.11}$$

Note that $\frac{\omega_i \cdot \omega_r}{2}$ is equivalent to $\omega_r \cdot N$ in this path, which is the expected input to evaluate the fresnel reflection in a typical surface reflection model.
Combining both terms, the scattering function for the R path becomes:

$$N_R(\phi) = F\left(\eta, \cos\frac{\omega_i \cdot \omega_r}{2}\right)\frac{1}{4}\cos\frac{\phi}{2} \tag{4.12}$$

Which is simple to compute. Note that ϕ is in the range $[0, \pi]$ because it refers to the minimum angle between two directions, therefore $\cos\frac{\phi}{2}$ is in the range $[0, 1]$, so there is no need to keep the absolute value from Eq. (4.10).

As suggested by Karis [Bri16], better performance can be achieved in this computation by using trigonometric identities to skip any inverse trigonometric function like $\arccos$ (which are usually expensive), given that we know $\cos\phi = \omega_{i\|} \cdot \omega_{r\|}$.

$$\cos\frac{\phi}{2} = \sqrt{\frac{1}{2} + \frac{1}{2}\cos\phi} \tag{4.13}$$

Figure 4.7 shows the contribution of the R path, using the same setup as in Figure 4.2. As expected, the R path does not contain any information of the hair color, and it maps approximately to an anisotropic specularity.

4.3.3 TT Path

The transmission path (double transmission, hence TT) accounts for energy that has entered the hair strand and has been transmitted forward to the outside (see Figure 4.6). For the TT path (with path index $p = 1$), we can begin by computing the single root h from Eq. (4.4), as done in Eq. (4.9). The outcome now is significantly more complex (obtained from d'Eon et al. [DFH+11]):

$$h_{TT} = \frac{\text{sign}(\phi)\cos\frac{\phi}{2}}{\sqrt{1 + a^2 - 2a\,\text{sign}(\phi)\sin\frac{\phi}{2}}}, \tag{4.14}$$

where $a = 1/\eta'$, and η' is the Bravais index of refraction (see Section 2.4.2.4). Karis [Bri16] then simplifies this again with the same trigonometric exchange strategy

Figure 4.7: Contribution of the R path ($p = 0$)

stated in Eq. (4.13):

$$h_{TT} = \frac{\frac{1}{2} + \frac{1}{2}\cos\phi}{1 + a^2 - 2a\sqrt{\frac{1}{2} - \frac{1}{2}\cos\phi}} \tag{4.15}$$

However, it is still a heavy computation, so he proposes an approximation of this term:

$$h_{TT} \approx (1 + a(0.6 - 0.8\cos\phi)\cos\frac{\phi}{2} \tag{4.16}$$

This expression has not been derived from any physical observation, it is just a cheaper function that the authors tuned to approximate the original one. Figure 4.8 shows the difference between the exact h and the approximated one, for different values of a.

It is important to be aware of these differences because there are scenarios in which the approximation will significantly fail, specifically for high a values, which map to low Bravais indices of refraction η'. If we take into account the Bravais index if refraction definition $\eta' = \frac{\sqrt{\eta^2 - \sin^2\theta_d}}{\cos\theta_d}$, η' is lower when the longitudinal difference angle θ_d is closer 0 (assuming $\eta \geq 1$). In this case, this maps to $\eta' = \eta$. Intuitively, this means that in this scenario there is no longitudinal difference between the incident and outgoing rays, therefore all the actual refraction happens in the cross-section of the fiber. Back to the approximation, this implies that the worst-case scenario is when a ray incides in a direction perpendicular to the hair

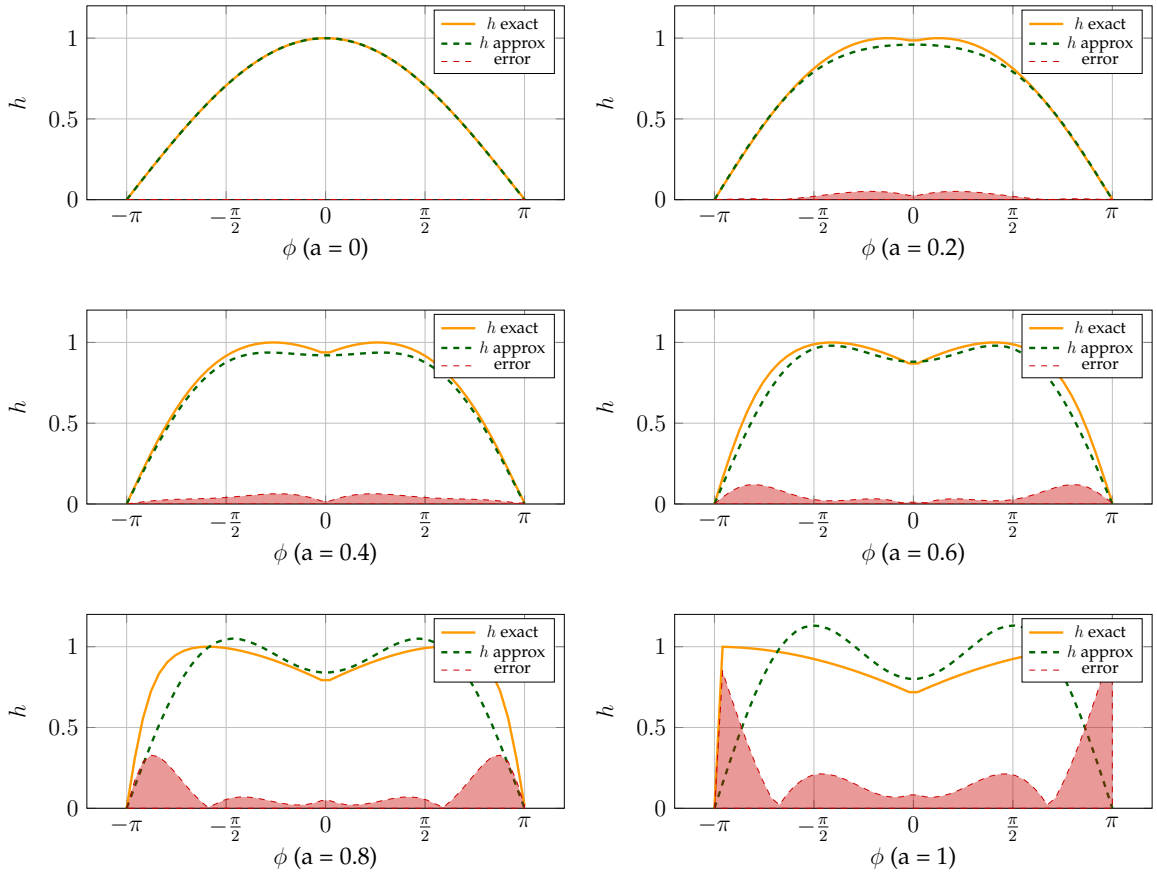


Figure 4.8: Comparison between the exact h and the approximated one used in the TT azimuthal path, for different values of $a = 1/\eta'$.

fiber, specifically when the incident and outgoing rays are in opposite directions (ϕ=π), and the index of refraction η is close to 1. Since we only are dealing with hair-specific refraction, which is around $\eta = 1.55$, the maximum possible a becomes $a = 1/\eta = 1/1.55 = 0.65$, which, as can be observed in Figure 4.8, already takes out most part of the error. Still, it is not favorable the fact of having a higher error when the incident and outgoing rays are in opposite directions, since these pairs usually contribute significantly to the TT path, and hence would require more accuracy.

Based on this, Tafuri [Taf19] focuses on approximating the actual analytical solution of the attenuation A, instead of only h, and ends up finding that $h = 0$ has a lower error in general than the approximation introduced above. In this work I have implemented both options, which allows one to check that indeed, using $h = 0$ achieves better results, especially for high absorption values, but Karis' achieves more pleasant results for low longitudinal difference angles θ_d and ab-

sorption values.

For the TT and TRT paths, both d'Eon and Marschner define the attenuation as a set of chained Fresnel reflections weighted by an absorption term. In this work I use Schlick's approximation [Sch94] for measuring the Fresnel F, which is responsible to account for the proportion of light reflected/refracted in every path.
As for the absorption, it accounts for the amount of light absorbed across the hair fiber in terms of the distance it has travelled (for more context, see Section 2.4.2.7). In the R path there is no absorption because the reflection is considered as a rough shifted mirror, but for TT and TRT, the light goes through the fiber, and therefore will get absorbed differently along its spectrum.
The absorption term has been described in many different ways in the literature, but for this work, I am using the approach by Pekelis et al [PHVL15]. This approach uses a color variable that maps implicitly to the absorption coefficients instead of using the absorption coefficients as raw input, which allows for a more intuitive tuning of the model:

$$T(\theta, \phi) = e^{-p\zeta(C)\left|\frac{\cos\gamma_t}{\cos\theta_d}\right|} \tag{4.17}$$

Then, substituting with $\gamma_t = \arcsin\frac{h}{\eta'}$ and developing for the TT path ($p = 1$):

$$T_{TT}(\theta, \phi) = C^{\frac{\sqrt{1-h^2a^2}}{2\cos\theta_d}}, \tag{4.18}$$

where C is the base color of the hair.
Note that in the case of Tafuri's approach, approximating A_{TT} at $h = 0$, most of the parameters involved in the attenuation can be made constant:

$$T_{TT}(\theta, \phi) = C^{\frac{1}{2\cos\theta_d}} \tag{4.19}$$

Also, from Eq. (4.7), f can be simplified:

$$f = F(\eta, \cos\theta_d) \tag{4.20}$$

This might slightly reduce the computational complexity of measuring this path, but not significantly, since at the end of the day we are only avoiding a few multiplications of already known parameters, not any complex operation.

Same as with the analytical h value, the analytical azimuthal distribution term for the TT path becomes much more complex compared to the one seen in the R path. Following the same line, Karis uses a cheaper approximation to that function.

Specifically, it uses a Gaussian with the appropriate parameters that produce a similar function to the analytical one. In this work, it has proved to be valid, so it is the one that is going to be used:

$$D_{TT}(\phi) \approx e^{-3.65\cos\phi - 3.98} \tag{4.21}$$

Then, the final expression for the azimuthal TT contribution is:

$$N_{TT} = (1 - F(\eta, \cos\theta_d))^2 C \frac{1}{2\cos\theta_d} e^{-3.65\cos\phi - 3.98} \tag{4.22}$$

In the same scene setup as in Figure 4.2, the TT path has almost no contribution in this (see Figure 4.9). This path becomes extremely significant when the hair strands are between the light source and the camera, as can be observed in Figure 4.10, which uses a point light behind the hair object (note that the intensity of this light is low), and even more prominently in Figure 4.17, which uses a directional light source facing the camera. Note that this path introduces slight information of the hair color due to the absorption inside the strand.

Figure 4.9: Contribution of the TT path ($p = 1$)

4.3.4 TRT Path

Similar to the previous paths, the goal is to find the right value(s) for h given ϕ, which will allow us to compute the attenuation and distribution terms. For the TRT path, as noted in Eq. (4.4) from Section 2.4.2.6, a given difference angle ϕ can

(a) TT ($p = 1$) (b) All components

Figure 4.10: TT path contribution placing a point light behind the head.

lead to either one or three solutions of h; the analytical solutions are the roots of the following function (extracted from [DFH+11]):

$$\sin\frac{\phi}{2} = -h + 2a^2h^3 + 2ah\sqrt{1-h^2}\sqrt{1-a^2h^2} \tag{4.23}$$

These solutions are complex and inconvenient to implement, which makes them unfeasible for real-time. Moreover, there is no cheap approximation for this solution even in the state-of-the-art of interactive hair rendering. Thus, I use the alternative introduced by Karis [Bri16], which is similar to the way Tafuri approximated h in the TT path [Taf19]. Basically, it consists in finding the h value that most contributes to the global attenuation function; Karis approximates it at $h = \frac{\sqrt{3}}{2}$. Consequently, the attenuation expression introduced by Pekelis [PHVL15] (see Eq. (4.17)) can be developed again, now with $h = \frac{\sqrt{3}}{2}$ and $p = 2$ (TRT), resulting in the following expression:

$$T_{TRT}(\theta,\phi) = C^{\frac{0.8}{\cos\theta_d}} \tag{4.24}$$

also, for the proposed constant value $h = \frac{\sqrt{3}}{2}$, γ_i becomes $\gamma_i = \pi_3$, so we can define:

$$f = F(\eta, \frac{\cos\theta_d}{2}) \tag{4.25}$$

In the same line, Karis approximates the distribution term with another Gaussian [Bri16]:

$$D_{TRT} = e^{17\cos\phi - 16.78} \tag{4.26}$$

So far, azimuthal roughness has not been taken into account for any of the paths. Note that the original Marschner approach does not take it into account, but recently Chiang et al. [CBTB16] demonstrate the relevance of this component for the resulting appearance.

For the R path, there is no need to consider it, since the light does not traverse the strand. For the TT path, Tafuri proposes to apply reconstructed Gaussians by fetching their parameters from a baked structure into a texture [Taf19]; this has not been implemented in this work in order to reduce the number of texture lookups as much as possible, since these are usually expensive.
Nevertheless, for the TRT path, Tafuri improves significantly the distribution approximation by using a scale factor s_r that depends on the azimuthal roughness β_N [Taf19]:

$$\begin{aligned} D_{TRT} &= s_r e^{s_r(17\cos\phi - 16.78)} \\ s_r &= \text{clamp}(1.5(1-\beta_N)) \end{aligned} \tag{4.27}$$

It is important to note that the scale factor s_r is not based on any physical measure; it is manually tuned to visually approximate the effect.
As can be observed in Eq. (4.27), adding the contribution of the azimuthal roughness β_N involves an almost negligible overhead in computational complexity with respect to the original form (only a small number of basic operations). Moreover, the TRT path is the one with the most significant azimuthal roughness contribution, since it is the path in which light performs more bounces inside the fiber. Because of these two reasons, I have chosen to use this improved approach in my implementation.

The resulting expression for the TRT path is:

$$N_{TRT} = \left(1 - F\left(\eta, \frac{\cos\theta_d}{2}\right)\right)^2 F\left(\eta, \frac{\cos\theta_d}{2}\right) C^{\frac{0.8}{\cos\theta_d}} s_r e^{s_r(17\cos\phi - 16.78)} \tag{4.28}$$

Analyzing visually the results, as can be observed in Figure 4.11, the TRT path is the one reflecting most part of the pure hair color, and its contribution is mainly diffuse.

4.3.5 Roughness

The roughness property introduced in the proposed hair models does not only influence the distribution of the light reflected, but it can also have an effect on the

Figure 4.11: Contribution of the TRT path ($p = 2$)

actual color of hair.

As noted by Chiang et al. [CBTB16], longitudinal roughness β_M primarily controls the width of the highlight, thus changing the perceived shininess of the fiber assembly, while azimuthal roughness β_N controls the overall softness.

Figure 4.12 shows the same setup with different longitudinal roughness β_M values. As can be observed, the influence of the longitudinal roughness on the output visualization can be mapped to the traditional roughness in a BRDF, in the sense that higher roughness values widen the distribution of reflected rays. Visually, this decreases the saturation of the hair color (assuming a non-colored light), since the pure color lobes are affected now by the R ad TT paths which add light-specific color information.

Figure 4.13 shows the influence of various azimuthal roughness values β_N. In this case, we can see how it has a more direct impact on the actual hair color. Note that it does not affect the R path, since the azimuthal roughness contributes to the interactions of light inside the hair strand.
The visible difference, in this case, is due the TRT path contribution. The azimuthal roughness is still modeling the microfacet geometry and therefore has an impact on the reflection distribution. With this in mind, we know that having a higher

(a) $\beta_M = 0.15$ (b) $\beta_M = 0.3$ (c) $\beta_M = 0.4$

Figure 4.12: Output visualization using different longitudinal roughness β_M.

roughness implies a wider distribution, so the outcoming light (which has been absorbed in a frequency-dependent way determining the hair color) gets scattered across many directions. Hence, the intensity scattered towards the camera will be lower than it would be if the azimuthal roughness was low, since it would then reflect/refract more directionally, transmitting more energy in the direction towards the camera. It is important to mention that this last scenario would happen only if the light and the camera are placed in such a way that the TRT path contribution is maximized. Otherwise, the outgoing radiance towards the camera will be lower than it would be with a higher roughness value (because most of the light will get scattered very directionally in another direction). In practice, this implies that moving either the camera or the light will change the color of the hair significantly, which is not what one would usually expect. Because of this, higher azimuthal roughness values are usually more appropriate for hair and fur.

4.4 Environmental Lighting

Applying environmental lighting via Image Based Lighting (IBL) in a BCSDF is not as straightforward as with traditional BRDF models. The main issue is that the BCSDF introduced cannot be approximated with an irradiance map as performed in IBL for traditional diffuse BRDF components. The reason why this happens is that most terms in the BCSDF need to remain inside the rendering equation integral since they depend on the differential variables. Thus, in principle, we cannot benefit from any precomputation. Evaluating the integral with the proposed

(a) $\beta_N = 0.3$ (b) $\beta_N = 0.7$ (c) $\beta_N = 0.95$

Figure 4.13: Output visualization using different azimuthal roughness β_N.

BCSDF numerically is far beyond feasibility in real-time applications, so we need to simplify.

There are some methods proposed in the literature that approximate the environment by a set of Spherical Radial Basis Functions (SRBF) [RZL+10], but again, their precision scales with numerical complexity.

To cope with this problem, Karis [Bri16] proposes to simply take a sample from the environment irradiance map in the direction we (heuristically) guess that will have the largest contribution to the hair appearance, and factor it properly to ensure the same amount of energy is taken into account. Karis [Bri16] uses a bent normal of the hair card to sample from the environment irradiance map. This work uses the actual surface normal instead, but the approach is the same: sampling from a precomputed irradiance map, and then factoring the sample by π, in order to compensate for the fact that we are not sampling from any other direction on the positive hemisphere, but is implicitly embedded in the irradiance map sample. This way, we get a sample that can be input into the shading model, ensuring that contiguous values will have a low variance in the IBL contribution. Furthermore, Karis' approach [Bri16] introduces other observation-based modifications:

- The R path contribution is multiplied by $min(\omega_i \cdot \omega_r + 1, 1)$. This translates to attenuating the reflections coming from the opposite direction to the camera (given this work's sampling strategy, this becomes $min(n \cdot v + 1, 1)$). The reason why this is done is because we cannot apply the shadowing techniques introduced in Section 4.5 to the IBL samples, since these are not actual

light sources, so we need to compensate for it by removing the intensity that would likely be blocked by the volume of hair.

- Because of the same reason, the whole TT path is removed, since it mainly contributes when the light behind the hair volume faces the camera.
- Increase the longitudinal roughness, approximating this way a larger area light source, which is appropriate when accounting for the diffuse environment lighting.

I have slightly modified this approach by assuming we are only dealing with hair cards. When fetching the normal direction used to sample the environment map, we allow for this normal to be inverted so that $n \cdot v > 0$, under the assumption that we are using double-sided polygons, and the face that the camera is seeing has more R and TRT contribution in its positive normal direction, which is not necessarily the case if we do not perform the conditional inversion proposed.
Note that this will implicitly solve the first point proposed by Karis, so with the new approach used there is no need to multiply the R path by any external factor.

Figures 4.14 and 4.15 show the contribution of IBL in an outdoor and indoor environment, respectively.

4.5 Shadowing

Shadowing in the context of hair rendering is important, and mandatory if we want to support dynamic scenes.

As explained in Section 4.1, the proposed implementation uses hair cards, which usually are placed in multiple layers through the hair volume. The scattering model is useful to highlight these differences in depth, so that the whole volume does not look flat and uniform. Despite this, in order to achieve a realistic sense of depth, we need to introduce shadows.

Because of the nature of hair cards, and the way we are blending them, traditional shadow maps cannot be used; if our shadow map consists in a basic depth map with the raw geometry projected into the light source view, once we test the depth of our point (being shaded from the main view) in light space against the shadow map, we might end up rejecting it as a false negative. This would happen if the shadow map point occluding our main view point was part of the cards

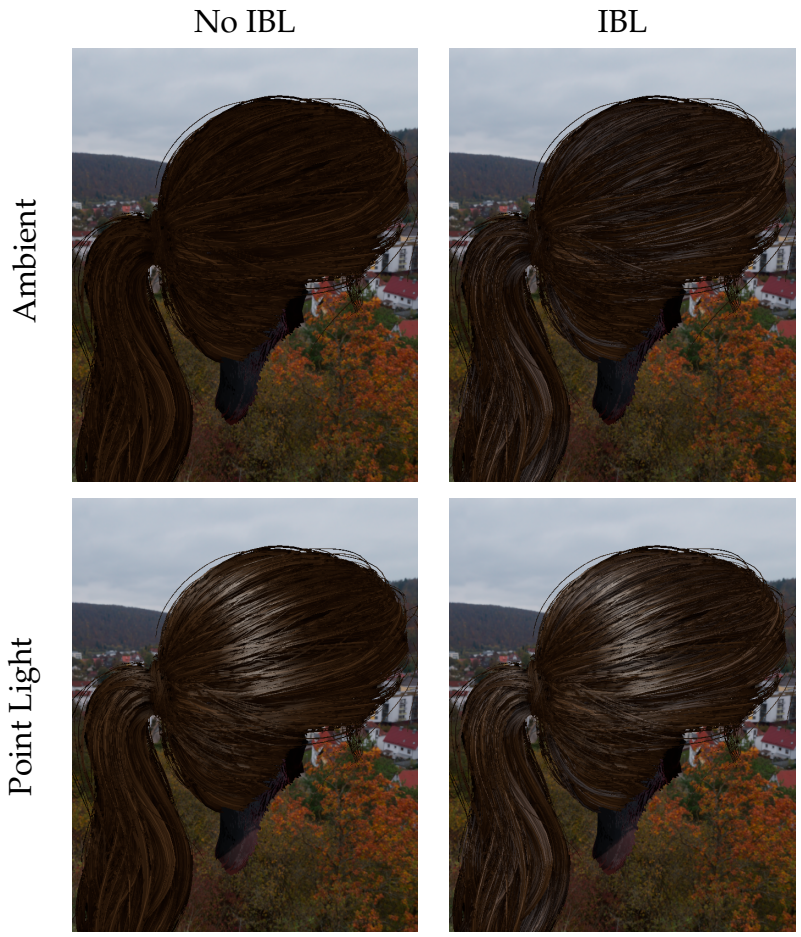


Figure 4.14: IBL influence (outdoor environment).

geometry but did not pass the alpha test (its corresponding alpha value from the opacity map is below the alpha threshold defined). Generalizing, our shadows would have the cards geometry shape instead of the strands shape.

A widely used solution for shadowing applied to blended objects is to evaluate the alpha test in the shadow map too. This implies having to use a custom pixel shader for performing this operation, which is the worst case from the hardware perspective to perform depth test; it cannot benefit from either *Early-Z* or *Z-Culling* (raster-time depth test) because it writes to the depth target directly from the pixel shader. For better quality, I also use percentage close filtering (PCF) [RSC87] with 4 taps (3x3 window) to smooth the shadows.

Figure 4.16 shows the contribution of shadows in the base scenario. As can be observed, shadows help to give hair a feel of volume with multiple layers. De-

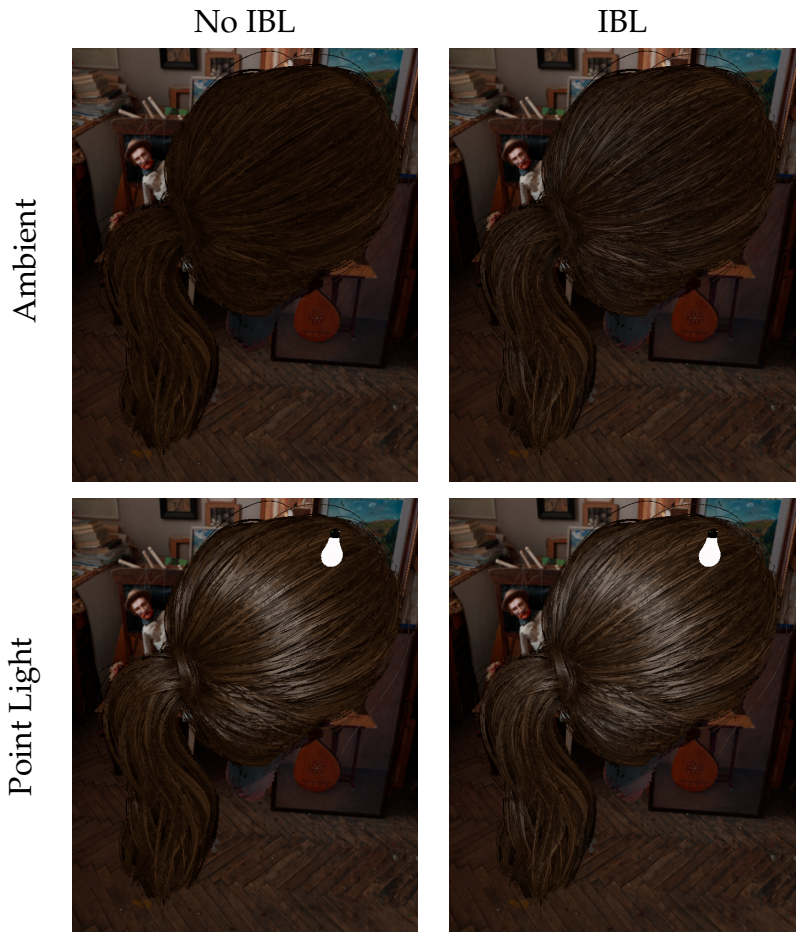


Figure 4.15: IBL influence (indoor environment).

spite the improvements, this is not enough for a proper representation of hair self-shadowing. As mentioned in previous sections, hair is translucent. This means that light entering a hair strand can still reach strands behind it, until it is absorbed completely. This is not modelled by the scattering model (since it only measures single-strand scattering), so we need to explicitly reproduce this effect.

For this work, I use the simple solution proposed by Karis [Bri16] which is efficient and feasible for real-time applications. It consists in modifying the current shadow map application by using an exponential falloff instead of a hard depth test, so that hair points behind the first lit strand can receive a positive amount of radiance from that light source, with this amount being inversely proportional to the distance (in light space) between the first lit strand and the hair point being shaded; the more distant, the lower the contribution from the light it will receive, which properly represents the absorption through the hair volume. Moreover, this

Figure 4.16: Shadowing comparison using a point light between the head and the camera. (left) shadows disabled, (right) shadows with exponential falloff.

extension is compatible with PCF, so I mix both.

Figure 4.17 shows an example of how relevant shadows are when we see the hair from the opposite side of the light source. Notice that light does not decay abruptly due to the exponential falloff contribution.

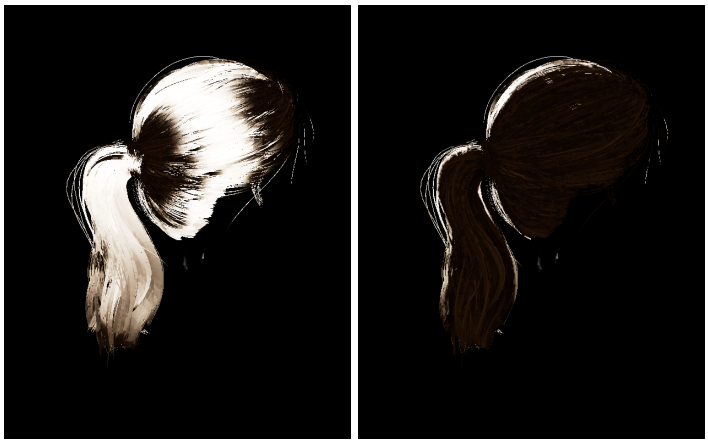


Figure 4.17: Shadowing comparison using a directional light oriented towards the camera. (left) shadows disabled, (right) shadows with exponential falloff.

4.6 Global Scattering

Global scattering through the hair volume is complex. Yan et al. [YSJR17] introduce a new BSSRDF for hair, which is convenient, but not feasible in real-time applications so far. Lately, a combination of *Dual Scattering* [ZYWK08] with *Deep*

Opacity Maps [YK08] has been used in real-time frameworks. Since my implementation does not use deep opacity maps to compute the shadowing, it is not worth going in this direction due to the additional computational cost it involves. Even if it was using *Deep Opacity Maps* for shadows, dual scattering is expensive and implies a complex implementation to measure efficiently the contribution of global scattering. Since frame times are already high by measuring the implementation from previous sections, I have chosen to simplify this component as much as possible.

It is important to understand which effect global scattering has on hair visualization. Ideally, global scattering is the contribution of light going through multiple strands inside the hair volume before leaving it, evaluating the scattering model for each of them. Therefore, the globally scattered light will have been partially absorbed according to the hair absorption coefficients multiple times before leaving the volume, which implies that it is going to contain information about the overall hair color. This is more noticeable in light-colored hair since it has a lower absorption. Furthermore, it varies smoothly across the hair volume.

With this in mind, I approximate it the same way global illumination is approximated in some real-time applications: a constant ambient term, which is the input hair color weighted by a user-defined factor, and the strand ID map; this map defines each baked strand with an 8-bit identifier (ID), which is used to attribute slightly different properties to the strands (like color), obtaining this way a realistic slight variance between strands.
Even if it is an oversimplification of the actual solution, I have observed that it has a remarkable contribution to the hair color in zones that are not affected by any of the scattering paths (R, TT, TRT). Figure 4.18 shows its contribution using the same scenario as in previous sections.

Figure 4.18: Contribution of the global scattering approximated by a constant ambient term.

Chapter 5

Experiments

5.1 Overall Strategy

A comparison with actual hair appearance, which would be the ground truth, is well beyond the scope of this work. This is because a comparison of the results with those of non-real-time rendering approaches does not seem a fair evaluation. The comparison which seems more appropriate is with alternative real-time approaches. However, it is very difficult to reproduce the models and approaches proposed in the literature, which are mostly unavailable. Thus, this chapter is focused on detailed comparisons with the current real-time approaches to simulate hair rendering based on the materials available in O3DE, on the basis of the hair model I have been using.

In the following section, I compare the quality of my results with the hair rendering with two different O3DE approaches: a *basic* PBR approach, and an *enhanced* PBR approach. I show that my results provide a better subjective hair appearance.

As the goal of this work was to achieve real-time performance, in the subsequent section, I discuss performance and scalability. I provide detailed performance and scalability results of my approach, with some of my components switched on and off, to be able to understand their cost more precisely. I show real time performance of my approaches, and I also compare the performance with the O3DE PBR models used for comparison in the second section. Furthermore, I evaluate the frame times of specific scene setups arranged in such a way that tests the limits and scalability of the model with respect to resolution, number of lights, and number of instances. I show that the proposed model scales almost like a basic

PBR in most cases, which is a positive result given the model complexity.

Some video demonstrations of the shading model can be found in this remote folder showing the real-time behavior of the implemented material with dynamic lighting and camera position.

5.2 Comparison with Basic and Enhanced PBR

In this section, I analyze the proposed shading model against the already implemented PBR model in O3DE. It is important to note that I do not have any ground truth result to compare against, mainly because of the complexity it would involve. Because of that, I describe the main observations to take into account for each comparison, and how these would relate to actual hair.

The PBR model used encompasses the most typical features used in real-time applications and has been tuned to achieve a good trade-off between efficiency and similarity to the real appearance of hair. More precisely, I have used properties like roughness, normal mapping, ambient occlusion, blending through alpha test, and anisotropic specularity (which achieves a directional highlight in the same fashion as the R path).

Figure 5.1 shows a side-to-side comparison of both my implemented material (right) and the PBR one (left) using the same scene setup.

Figure 5.1: Comparison between the hair material implementation and O3DE's PBR implementation tuned to achieve efficiency and similarity. Left: O3DE's PBR. Right: proposed hair material.

Even though the PBR model (Fig. 5.1, left) achieves a fair overall specular highlight similar to what we would expect in real hair, it can be observed that it cannot reach the precision of the hair material proposed (Fig. 5.1, right). This statement has been confirmed by experts in the field as well, in an informal and qualitative way.
This result is explained by the fact that the O3DE's PBR material uses a BRDF instead of a hair-specific BCSDF, while in my material I use a proper BCSDF.

In Figure 5.2, the same comparison is proposed but in this case enhancing the PBR model to achieve maximal quality without caring about performance. This involves the use of additional properties like screen-space subsurface scattering to obtain a TRT-similar contribution, multiple scattering for the specular bounce, parallax to accentuate different layers of depth, and PCF shadows. These techniques are already implemented in O3DE.

Figure 5.2: Comparison between the hair material implementation and O3DE's enhanced PBR implementation tuned to achieve maximal quality. (left) O3DE's enhanced PBR, (right) Implemented hair material.

The enhanced PBR achieves much better results than the basic one, and according to the experts questioned, it seems to evaluate the specularity with more detail, probably due to the parallax effect on the normals. Despite this, it still fails in accomplishing a noise-free and coherent reflection, as asserted by the experts as well, which makes my implementation to look more realistic as a whole.
The color contribution due to the back scattering on the enhanced PBR material (which would map to the TRT path in the proposed hair material) appears to exhibit higher similarity to real hair than the basic PBR one, but it still fails at some specific points like the ponytail, or points that have been shadowed wrongly like

the part at the bottom of the hair, which is alleviated in the proposed implementation with the exponential falloff.

Even though the reference cannot be recreated, Figure 5.3 shows a side-to-side comparison between Karis' hair shading model and this work's model under a similar scene and lighting conditions.

Figure 5.3: Comparison between Karis' hair material against the material implemented in this work. (left) Karis hair [Bri16], (right) this work´s hair material.

The first thing we might notice is the difference in the quality of the hair model. Karis' seems to have higher resolution and to be better authored, so the individual strands baked can be better perceived. Also, the strand ID map of the model I use is composed of a single channel, which limits the variability of strand color to be used and is probably not the case in Karis' model. The opacity is better solved in Karis' model, most likely because the maps used have a better resolution, the blending techniques are more elaborated, and on top of that, it has been antialiased properly. Aside from that, the actual shading looks similar. Since the hair color is dark, the lack of azimuthal roughness on the Karis' model is not very visible in this comparison.

5.3 Scalability and Performance

In order to evaluate the feasibility of the proposed model for real-time applications, we need to evaluate its performance.
Figure 5.4 plots the frame time of the three models presented in this chapter (basic PBR, enhanced PBR, and my hair-specific model) against different setups, for each one building on top of the previous one:

- **Raw hair:** The shading model with blending only.
- **Shadows:** Raw hair with shadows enabled.
- **IBL:** Raw hair with shadows and IBL enabled.

The timing measurements have been captured with *NVidia Nsight* on a *RTX3080* GPU, *Intel i7-11850H* CPU, and 32GB of RAM. The captures have been done in the O3DE Editor with a scene viewport resolution of 1494x835. This is a higher resolution than *HQ* (720p) and lower than *FHD* (1080p).

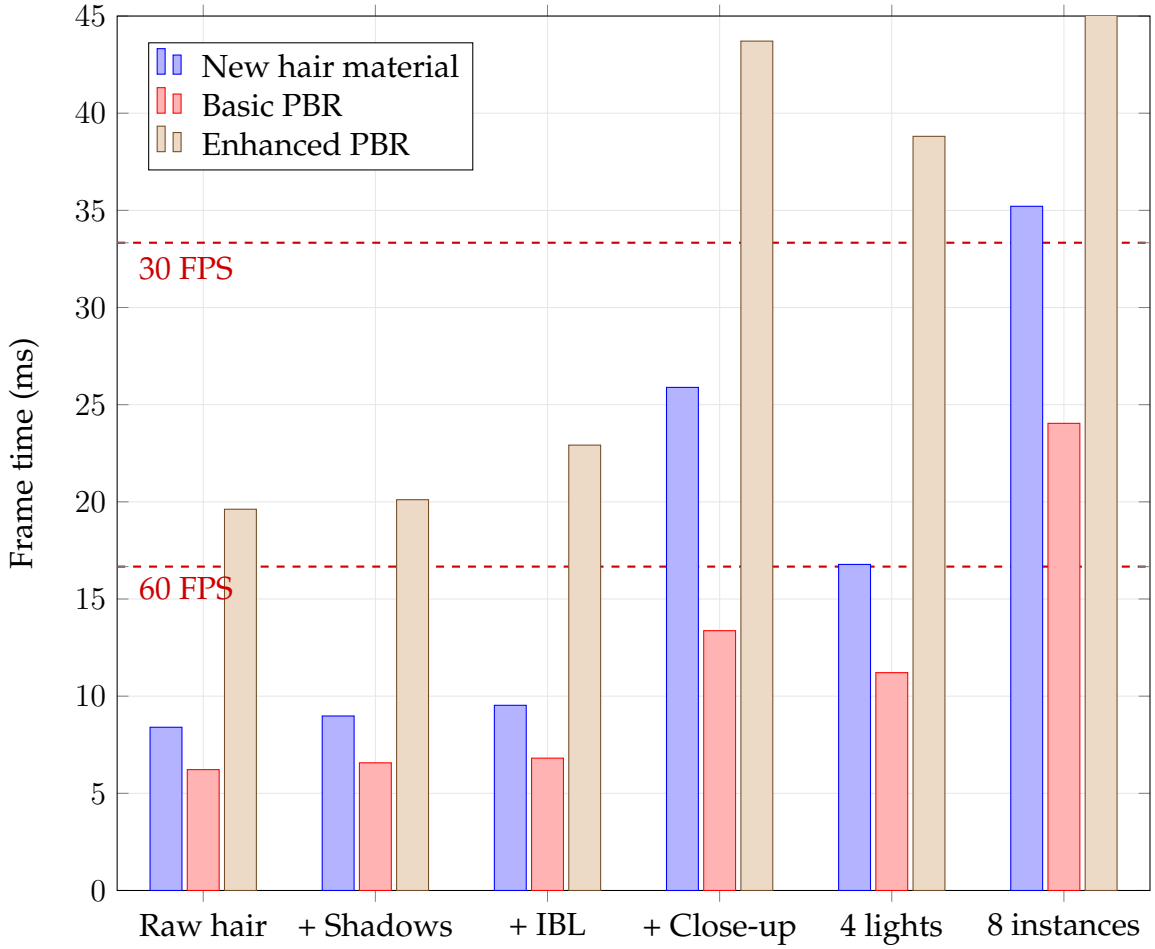


Figure 5.4: Frame times of each shading model introduced in the comparison for different scene setups.

In general, we can see how the hair material introduced is slower than the basic PBR but outperforms the enhanced PBR in all cases. This is already a positive

result demonstrating that the new hair material can achieve better results with better performance than a PBR replica of it would. The complexity and quality of the basic PBR are much lower, so we could expect it to be more efficient.

Apart from this per-component analysis, it is relevant to acknowledge the limits of the proposed material and how well it scales. Because of this, I have added three setups that could be considered stress tests for our model and are described in the following sections.

In general, we can see that in non-stress tests our method is able to shade at 60 FPS, and around 30 FPS in the worst stress tests, making it feasible for most real-time uses.

In all the experiments, the scene set-up (lights, environment, etc) described in Section 4.1 has been used equally unless noted otherwise. The implemented model is evaluated with disabled IBL unless noted otherwise.

5.3.1 Resolution

To see how the hair material scales with resolution, I use a close-up view of the hair model, since in this case most part of the screen is going to be occupied by hair that needs to be shaded with the given model on a per-pixel basis. This experiment is done with IBL enabled.

As we could expect, Figure 5.4 (*Close-up*) shows that the current implementation scales significantly with resolution. Every screen pixel needs to evaluate the complex shading model, which involves computing the contribution of each scattering path with its pertinent cost. This could be optimized by improving the shader implementation, targeting a better thread scheduling and synchronization strategy on the GPU side. The way to optimize it would be to look for potentially divergent branches that may affect threads from the same warp (usually neighbor pixels), and replace them with GPU-friendly alternatives. Also, profiling meticulously would help to identify these caveats, and reduce the vector registers (VGPR) pressure on the GPU. This experiment increases the cost of the basic and enhanced PBR too, but by a lower factor.

5.3.2 Light instances

To see how our model scales with the number of light sources, I have used four light sources in the same scene, as Figure 5.5 shows.

We can observe in Figure 5.4 (*4 lights*) how our model scales with lighting worse than the basic PBR. The reason behind this is that each scattering path contribution needs to be evaluated for each light source, plus perform the proper shadow map with pixel shader for every light as well. A solution to this could be to cluster light sources so that the model is only evaluated with a reduced amount of lights, at the cost of losing precision.

Figure 5.5: Output visualization of experiment with 4 discrete light sources.

5.3.3 Hair instances

This experiment measures how the proposed implementation scales with the number of hair instances. The screen occupancy is similar to the one in the close-up test, and two light sources have been used. The scene used is shown in Figure 5.6.

Just putting together the increased screen occupancy and the increased number of light sources is going to increase the frame rendering time significantly. On top of that, note that there is a significant amount of occlusion among instances. Since this work uses alpha testing and does not ensure the order in which the instances are being rendered, there is going to be a significant amount of overdraw. The performance decrease due to this can be seen in Figure 5.4 (*8 instances*). A solution to

this could be to sort the instances in a front-to-back order and draw separately (instead of GPU instanced) every instance with the goal of benefiting from the depth buffer that the previous rendered instance will have written into, so that occluded fragments can be discarded instantly.

In this case, the overdraw happens as well in the basic and enhanced PBR, but since there is an increase in screen occupancy and number of lights as well, the performance scaling factor with respect to the baseline measure is lower than in the proposed model.

The next step to improve the scaling in the proposed framework would be to implement a proper LODding mechanism, not in the hair geometry, since the vertex throughput is not the bottleneck in this case, but in the shading model instead, so that if an instance is far enough, a simplified model (or even the basic PBR introduced in this section) could be used.

Figure 5.6: Output visualization of experiment with 8 hair instances occupying most part of the viewport and 2 discrete light sources.

Chapter 6

Conclusions

In this work, a new framework for real-time hair rendering has been proposed as a combination of solutions for the different challenges discussed in the introduction. I briefly recall them here, with the solutions proposed.

Light scattering in single hair strands is solved with an approximation of previous physically-based models introduced in the film industry. This approximation is decomposed into several components accounting for the contribution of each light path at a strand level, where each piece has been defined based on the current real-time physically-based hair rendering state-of-the-art. The contribution of this work is to put together the best option for each component from the set of real-time references, where *best* is considered in terms of quality-performance ratio. Hair self-shadowing is performed efficiently with an extension of current shadow mapping techniques, and similarly for blending. Traditional techniques cannot be used because we need to account for a continuous and soft attenuation inside the hair volume. Finally, global scattering inside the hair volume is accounted as a constant ambient contribution, which has been observed to be a reasonable simplification qualitatively.

The proposed solution is able to render multiple hair instances in real-time, with similar quality to the state-of-the-art on interactive hair visualization. Despite this, there exist multiple areas of improvement; A proper LODding system on the shading complexity (not the geometry) would allow to display a higher number of hair instances on screen with similar perceived quality. Furthermore, self-shadowing and global scattering in hair could be solved with volumetric approaches. Experiments with such techniques would be needed, but these could

potentially lead to a better performance at the cost of a higher GPU memory footprint (assuming a full GPU implementation). Also, blending could be improved with alpha to coverage or a proper order-independent transparency implementation.

Apart from combining previous existing methods for hair rendering, this work introduces a framework for easier inspecting and interacting with the outcome. It contains a proper user interface so that the parameters input into the model can be modified interactively. Furthermore, it is modular, allowing for quickly enabling, disabling, or weighting each component involved in the model. This has permitted to perform proper comparisons and visualize efficiently the contribution of each component. It is important to mention that the development of this framework has involved a significant amount of time, reducing the time available for the actual hair model implementation. Despite this, the fact of having a framework allows to work on the actual models faster, in a more scalable way, and easier to extend. It also increases intelligibility and replicability.

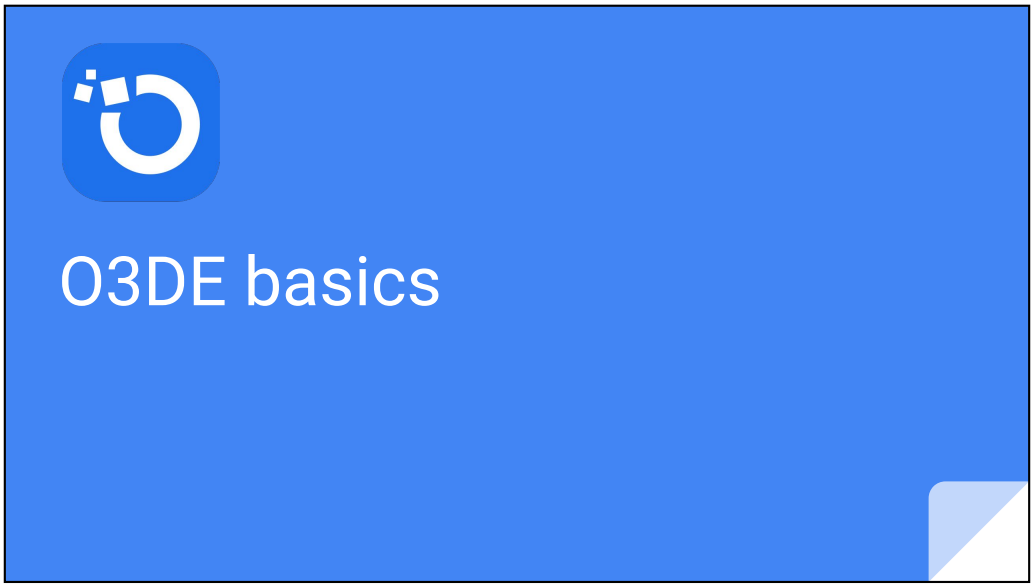
The framework has been developed in O3DE, an emergent promising open-source 3D engine, but it is worth to mention that currently it is not as stable and performant as other options available, therefore a small plus of performance could be expected (as of today) from porting the framework to other engines.

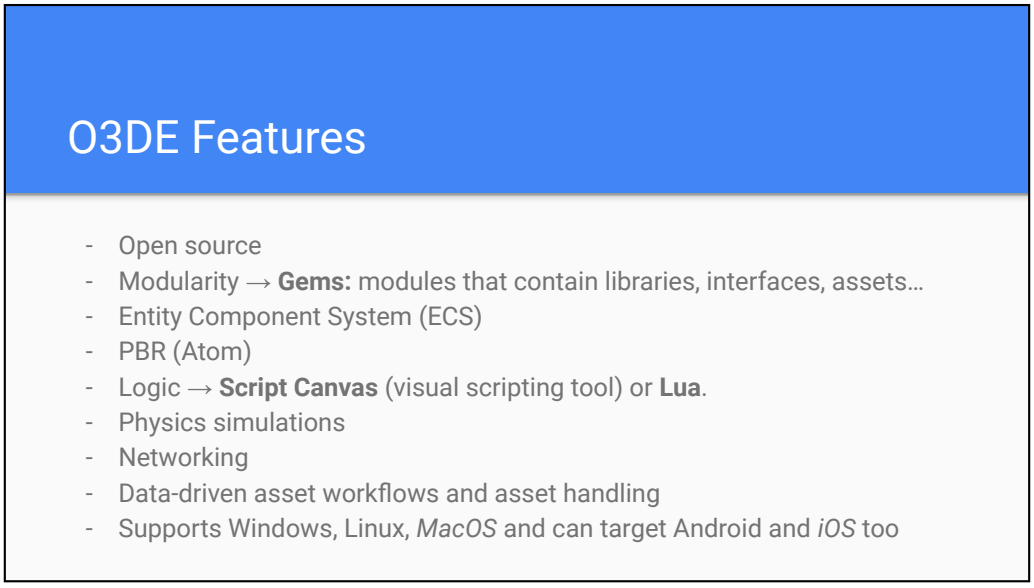
Appendix A

O3DE Background

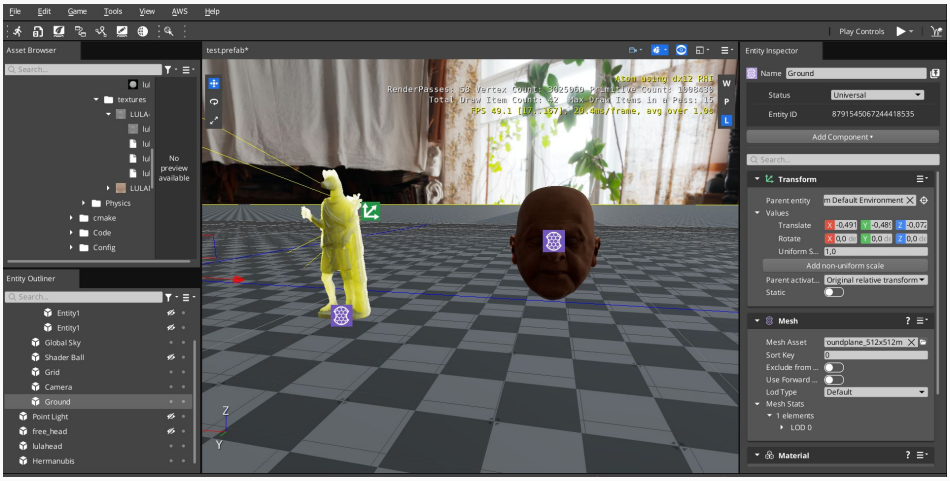
O3DE is an emerging open-source 3D engine. In this appendix, a set of slides are attached presenting the basic workflow of the engine and its rendering pipeline in a visual way. Afterwards, a document summarizing the components of O3DE's renderer *Atom* is presented, including how the pipeline and hardware interface work internally and an overview of the pass system, based on the talk *Atom Deep-Dive* from *O3DECon 2021*.

A.1 O3DE Basics

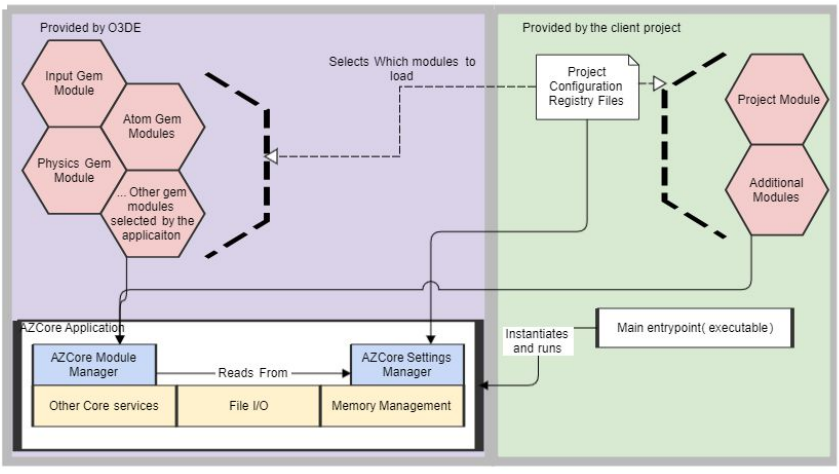

O3DE basics


O3DE Features
- Open source
- Modularity → **Gems:** modules that contain libraries, interfaces, assets...
- Entity Component System (ECS)
- PBR (Atom)
- Logic → **Script Canvas** (visual scripting tool) or **Lua**.
- Physics simulations
- Networking
- Data-driven asset workflows and asset handling
- Supports Windows, Linux, *MacOS* and can target Android and *iOS* too

Editor overview



Structure of project runtimes in O3DE [source]



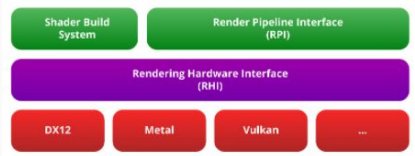
Atom renderer interfaces

Render Pipeline Interface (RPI)

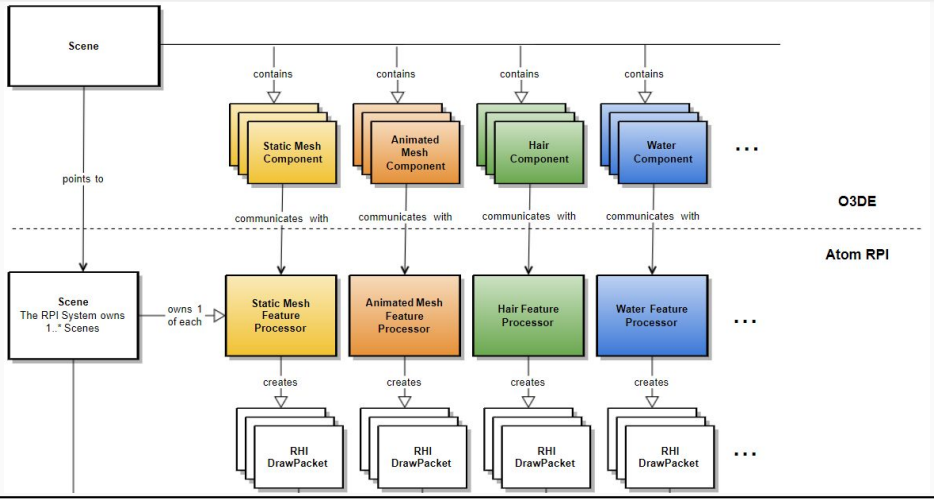
Main interface for developers to program the render pipeline.

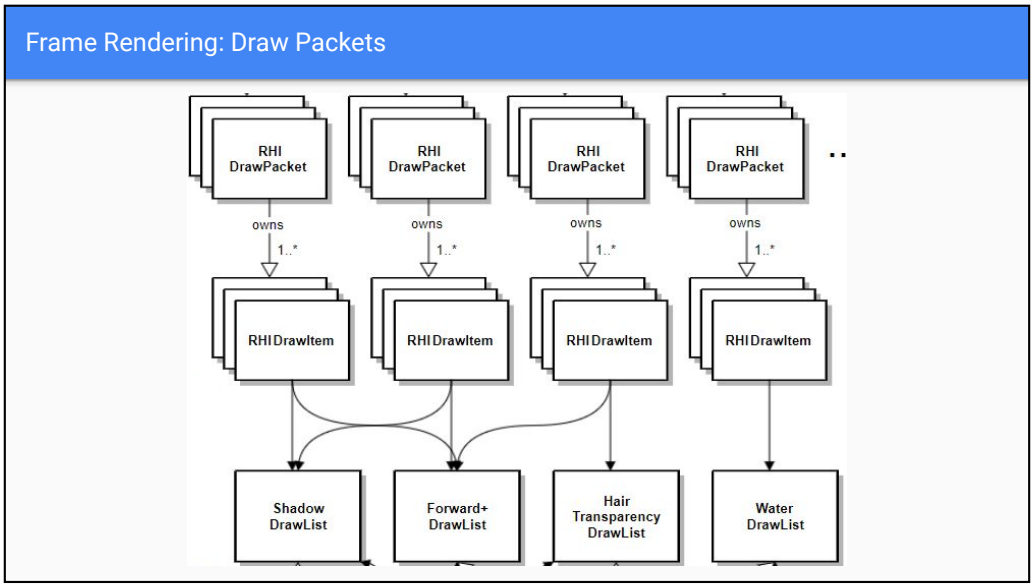
Rendering Hardware Interface (RHI)

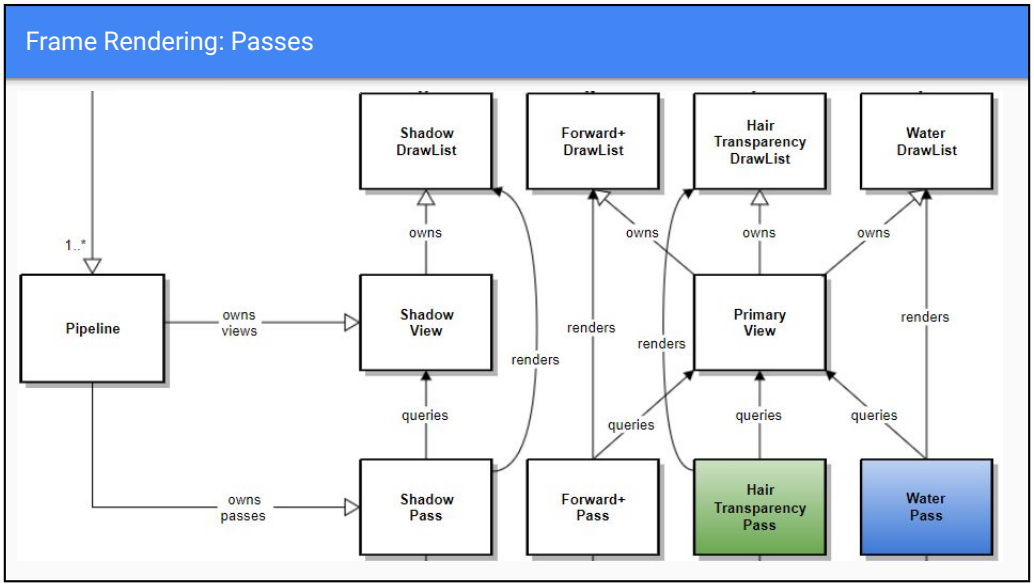
Low-level interface that abstracts platform-specific code.

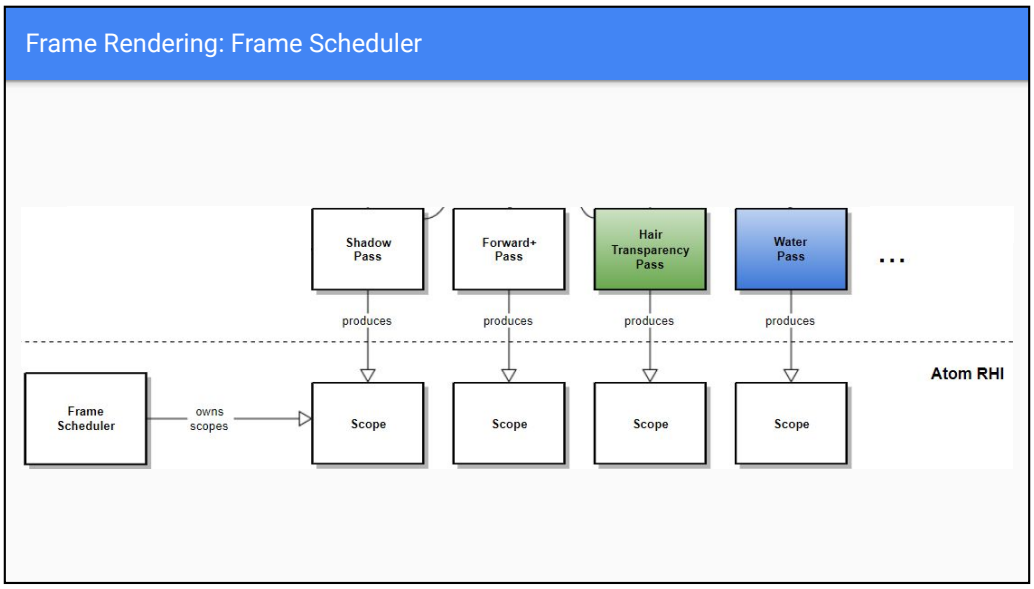


Frame Rendering [source]




Frame Rendering: Draw Packets
RHI DrawPacket
RHI DrawPacket
RHI DrawPacket
RHI DrawPacket
owns
1..*
RHI DrawItem
Shadow DrawList
Forward+ DrawList
Hair Transparency DrawList
Water DrawList


Frame Rendering: Passes
Pipeline
owns views
owns passes
Shadow View
Shadow Pass
Forward+ Pass
Primary View
Hair Transparency Pass
Water Pass
Shadow DrawList
Forward+ DrawList
Hair Transparency DrawList
Water DrawList
owns
queries
renders


Frame Rendering: Frame Scheduler
Shadow Pass
Forward+ Pass
Hair Transparency Pass
Water Pass
...
produces
produces
produces
produces
Atom RHI
Frame Scheduler
owns scopes
Scope
Scope
Scope
Scope

A.2 O3DECon 2021: Atom Deep Dive - Talk summary

A.2.1 Introduction

Atom is the renderer used by O3DE. In this talk, some of its main developers (Anton Michels and Sidharth Moudgil) explain how it works internally.
Atom is divided in two parts, the Render Pipeline Interface (RPI) and the Render Hardware Interface (RHI), which are explained in the following sections.

A.2.2 RPI

The RPI's job is to get draws from the application to the RHI. For example, a material can have multiple shaders, used for various aspects of rendering. So, for each shader associated, the RPI generates a *DrawItem*. The generated set of *DrawItems* is bundled into a *DrawPacket*, which contains all the data needed to render the material (using a mesh, volume...) in all views and passes. Each shader needs to have an associated pass, which is achieved by using *DrawListTags*, unique tags defined per pass.

A.2.3 Pass System

A pass is a collection of GPU work describing the logic of the renderer, with slots for inputs and outputs.

Atom supports C++ and Data Driven pass authoring, but also both paradigms work seamlessly together. The data-driven approach (using a JSON file) provides speed (no compilation needed) and usability, while defining it in C++ offers more control and customization. To define the data to fill each slot (inputs and outputs), atom uses *attachments*. *Attachments* form connections between passes, and like most graphics API define, these can be *image attachments* or *buffer attachments*.

Passes can contain other passes, which allows to build a pass tree hierarchy (with the root of the tree being a root pass, which must be a parent pass). Each pass is defined via *Pass Template* and can be invoked via *Pass Request*, which instantiates a *Pass Template* inside a parent pass.
In the talk, an actual pass is shown and examined as an example using a JSON file authoring method. It illustrates the connections and requests formatting, and how

to link attachments between related passes. The pass file is structured as shown in Figure A.1.

```
{
    "Type": "JsonSerialization",     // JSON Serialization Header
    "Version": 1,
    "ClassName": "PassAsset",
    "ClassData": {
        "PassTemplate": {            // PassTemplate
            "Name": "DownsamplePassTemplate",
            "PassClass": "ComputePass",
            "Slots": [ ... ],
            "PassRequests": [ ... ],
            "ImageAttachments": [ ... ],
            "Connections": [ ... ],
            "FallbackConnections": [ ... ],
            "PassData": { ... },
            }
    }
}
```

Figure A.1: Pass file structure

A.2.4 RHI

The RHI is an interface layer abstracting the calls for each specific modern graphics API (DirectX12, Vulkan, Metal, console...), so that the developer can still use low-level graphics features that the APIs offer, but with cross-API compatibility, which is fundamental to achieve a cross-platform engine.

The RHI is also in charge of placing synchronization mechanisms between passes automatically and using resources as efficiently as possible. To perform this, they first build a frame graph, which is a directed acyclic graph of all the render passes and the resources in a frame. The frame graph contains information about inter-pass dependencies.

To further optimize the GPU performance, some improvements are added:

1. **Efficient resource binding at different frequencies:** Relates to the Shader Resource Groups (SRG) compilation stage. A SRG represents the cooked data that is committed to the GPU in the Compile phase. Each grup represents a different update frequency (e.g. once per frame, once per material...). Using this concept of SRG, it is easy to internally implement the binding in each specific API, since in most of them it is managed similarly (descriptor sets, tables...).

2. **Track explicit resource dependency:** From the frame graph information, resources and scopes can be prepared to compile with necessary synchronization mechanisms.

3. **Transient Memory Aliasing:** Transient memory refers to resources (buffers or images) that are only valid in the current frame. The goal is to use memory aliasing whenever possible; Memory aliasing happens when two different resources use the same memory space. It is something we want to use when a resource is not going to be used for the rest of the frame, but we still need to create new ones. Then, keeping the space reserved for the not used resource would be a waste of memory.

4. **Encoding parallelism:** Dividing thread work by scopes/nodes in the frame graph can lead to long stalls because a node might have much more work involved than the other ones. Therefore, they propose to divide based on some of the following criteria:

   - Cost Threshold: Amount of work involved.
   - Swapchain: Depends on the required framebuffers.
   - Async Queue: Check if the work comes from a different queue.
   - Fence boundary: Amount of synchronization mechanisms in the boundaries.

   In any case, group and context ordering must be preserved, so if there is dependency between groups, synchronization mechanisms should be added.

Bibliography

[Bri16] Brian Karis. Physically Based Hair Shading in Unreal. *Siggraph*, 2016.

[CBTB16] Matt Jen Yuan Chiang, Benedikt Bitterli, Chuck Tappan, and Brent Burley. A Practical and Controllable Hair and Fur Model for Production Path Tracing. *Computer Graphics Forum*, 35(2):275–283, May 2016. Publisher: John Wiley & Sons, Ltd.

[DFH+11] Eugene D'eon, Guillaume Francois, Martin Hill, Joe Letteri, and Jean-Marie Aubry. An Energy-Conserving Hair Reflectance Model. *Eurographics Symposium on Rendering*, 30(4), 2011.

[Gol17] Ben Golus. Anti-aliased Alpha Test: The Esoteric Alpha To Coverage, 2017.

[KN01] Tae-Yong Kim and Ulrich Neumann. Opacity Shadow Maps. In Steven J. Gortler and Karol Myszkowski, editors, *Rendering Techniques 2001*, pages 177–182. Springer Vienna, Vienna, 2001. Series Title: Eurographics.

[MJC+03] Stephen R Marschner, Henrik Wann Jensen, Mike Cammarano, Steve Worley, Worley Laboratories, and Pat Hanrahan. Light Scattering from Human Hair Fibers. 2003.

[PHVL15] Leonid Pekelis, Christophe Hery, Ryusuke Villemin, and Junyi Ling. A Data-Driven Light Scattering Model for Hair. 2015.

[PJH16] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann, September 2016. Google-Books-ID: iNMVBQAAQBAJ.

[RSC87] William T. Reeves, David H. Salesin, and Robert L. Cook. Rendering antialiased shadows with depth maps. In *Proceedings of the 14th annual*

conference on Computer graphics and interactive techniques, SIGGRAPH '87, pages 283–291, New York, NY, USA, August 1987. Association for Computing Machinery.

[RZL+10] Zhong Ren, Kun Zhou, Tengfei Li, Wei Hua, and Baining Guo. Interactive Hair Rendering Under Environment Lighting. page 8, 2010.

[Sch94] Christophe Schlick. An Inexpensive BRDF Model for Physically-based Rendering. *Computer Graphics Forum*, 13:233–246, 1994.

[Taf19] Sebastian Tafuri. Strand-based Hair Rendering in Frostbite. 2019.

[Tok18] Kirill Tokarev. Creating Hair for Real-Time Game Characters, January 2018.

[Wei06] Xue Wei. What is human hair? a light and scanning electron microscopy study, 2006.

[WM17] Chris Wyman and Morgan McGuire. Hashed alpha testing. In *Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pages 1–9, San Francisco California, February 2017. ACM.

[YJR17] Ling Qi Yan, Henrik Wann Jensen, and Ravi Ramamoorthi. An efficient and practical near and far field fur reflectance model. *ACM Transactions on Graphics (TOG)*, 36(4), July 2017. Publisher: ACM PUB27 New York, NY, USA.

[YK08] Cem Yuksel and John Keyser. Deep Opacity Maps. *Computer Graphics Forum*, 27(2):675–680, April 2008.

[YSJR17] Ling Qi Yan, Weilun Sun, Henrik Wann Jensen, and Ravi Ramamoorthi. A BSSRDF model for efficient rendering of fur with global illumination. *ACM Transactions on Graphics*, 36(6), November 2017. Publisher: Association for Computing Machinery.

[YTJR15] Ling-Qi Yan, Chi-Wei Tseng, Henrik Wann Jensen, and Ravi Ramamoorthi. Physically-accurate fur reflectance: modeling, measurement and rendering. *ACM Transactions on Graphics*, 34(6):1–13, November 2015.

[ZYWK08] Arno Zinke, Cem Yuksel, Andreas Weber, and John Keyser. Dual scattering approximation for fast multiple scattering in hair. In *ACM SIGGRAPH 2008 papers on - SIGGRAPH '08*, page 1, Los Angeles, California, 2008. ACM Press.